UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

FÁBIO MÁRCIO TAVARES LIMA

ESTUDO COMPARATIVO ENTRE TRÊS TECNOLOGIAS DE TELAS MULTITOQUE: ENFOQUE EM HARDWARE E SOFTWARE

São Luís
2013

FÁBIO MÁRCIO TAVARES LIMA

ESTUDO COMPARATIVO ENTRE TRÊS TECNOLOGIAS DE TELAS MULTITOQUE: ENFOQUE EM HARDWARE E SOFTWARE

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Carlos de Salles Soares Neto

São Luís
2013

Lima, Fábio Márcio Tavares.

Estudo comparativo entre três tecnologias de telas multitoque: enfoque em hardware e software/ Fábio Márcio Tavares Lima. – São Luís, 2013.

60f.

Impresso por computador (fotocópia).

Orientador: Carlos de Salles Soares Neto.

Monografia (Graduação) – Universidade Federal do Maranhão, Curso de Ciência da Computação, 2013.

1. Tela multitoque. 2. Software. 3. Hardware. I. Título.

CDU 004.354.3

FÁBIO MÁRCIO TAVARES LIMA

ESTUDO COMPARATIVO ENTRE TRÊS TECNOLOGIAS DE TELAS MULTITOQUE: ENFOQUE EM HARDWARE E SOFTWARE

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Aprovada em: 07 / 03 / 2013.

BANCA EXAMINADORA

Carlos de Salles Soares Neto

Prof. Dr. Carlos de Salles Soares Neto (Orientador)
Doutor em Informática
Universidade Federal do Maranhão

Maria Auxiliadora Freire

Prof.ª Msc. Maria Auxiliadora Freire
Mestre em Engenharia Civil
Universidade Federal do Maranhão

Portela

Prof. Msc. Carlos Eduardo Portela Serra de Castro
Mestre em Informática
Universidade Federal do Maranhão

Aos meus pais, Raimundo Alexandrino de Sousa Lima e Sônia Maria Tavares Lima.

AGRADECIMENTOS

A Deus, que me deu forças e esperança para superar tantos obstáculos e alcançar mais esta meta.

Aos meus pais, Raimundo Alexandrino de Sousa Lima e Sônia Maria Tavares Lima, por sempre acreditarem em mim e na minha capacidade.

Ao meu tio, Otílio Sousa Lima, pelo primeiro passo dado.

Ao meu orientador, Carlos de Salles Soares Neto, pelo apoio, desafio do tema, orientação e paciência dispensada.

Ao coordenador do Curso de Ciência da Computação, Carlos Eduardo Portela Serra de Castro, pela paciência e pela resolução célere dos assuntos burocráticos e administrativos.

A Natália Fernanda Gaspar de Almeida, pela ajuda na normatização e pelo encorajamento nas horas mais complicadas.

A Manoel Cavalcanti Filho, pelo conhecimento e pelos contatos nas horas certas.

E a todas as pessoas que acreditaram e torceram por mim em todos esses anos de afastamento acadêmico e no que esta conquista traz de positivo à minha vida.

“Tudo o que temos de decidir é o que fazer com o tempo que nos é dado.”
Gandalf The Gray

RESUMO

Estudo de revisão bibliográfica para comparação de tecnologias de telas multitoque. As telas multitoque são fundamentais para a facilitação da interação homem/máquina e as mesmas possuem diferentes componentes de software e hardware e o comparativo das mesmas serve para esclarecer qual possui melhor custo-benefício. Após análise de componentes básicos dos equipamentos PQ Labs™, Microsoft®Surface 1.0 e Microsoft®PixelSense, projetou-se um modelo de aplicativo baseado em bibliotecas de gestos e conclui-se que o equipamento da PQ Labs™ é o mais indicado devido à facilidade de implementação e menor custo.

Palavras-chave: telas multitoque, PQ Labs™, Microsoft®Surface 1.0, Microsoft®PixelSense, software, hardware.

ABSTRACT

Study literature review to compare multitouch screens technologies. The multitouch screens are key to facilitating the interaction man/machine and they have different hardware and software components and the comparison of them serves to clarify what has best value for money. After analyzing the basic components of equipment PQ Labs™, Microsoft®Surface 1.0 and Microsoft®PixelSense, was designed an application model based in libraries of gestures and concluded that the equipment from PQ Labs ™ is the most suitable due to ease implementation and lower cost.

Keywords: multitouch screens, PQ Labs™, Microsoft®Surface 1.0, Microsoft®PixelSense, software, hardware.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

CES	–	Consumer Electronics Shows
CIR	–	Collaborative Information Retrieval
CPU	–	Central Processing Unit
DI	–	Diffused Illumination
DSI	–	Diffused Surface Illumination
FTIR	–	Frustrated Total Internal Reflection
GPU	–	Graphics Processor Unit
HD	–	High Definition
IHC	–	Interação Homem/Computador
IR	–	Infravermelho
LAN	–	Local Area Network
LED	–	Diodo Emissor de Luz
LLP	–	Laser Light Plane
SDK	–	Software Development Kit
TM	–	Tela Multitoque
TUIO	–	Tangible User Interface Object
USB	–	Universal Serial Bus
WPF	–	Windows Presentation Foundation

SUMÁRIO

1 INTRODUÇÃO

Para Hix e Hartson (1993), a área de IHC (interação homem/computador) é responsável pelo estudo da forma com que homem e máquina interagem. Tem-se em seu escopo, a interface de software e de hardware com o usuário, a modelagem dos sistemas, o estudo da cognição e comportamento humano, estudos empíricos, metodologias, técnicas e ferramentas. O objetivo, na maior parte do tempo, é garantir um bom grau de usabilidade ao usuário.

Como visto em Shneiderman (2005), sob todo o tema de IHC existe a ideia do foco nas pessoas, nos usuários. Suas necessidades, capacidades e preferências para conduzir diversas atividades devem dar norte aos desenvolvedores ao construir uma interface. As pessoas não devem mudar a maneira com que interagem com um sistema para adaptarem-se a ele. Ao invés disso, o sistema deve ser construído para preencher os anseios dos usuários.

Segundo Han (2012) multitoque é uma tecnologia de IHC que consiste em uma tela sensível ao toque que reconhece múltiplos contatos que são simultaneamente interpretados pelo sistema, possibilitando vários usuários interagirem com o mesmo computador. Uma tela multitoque (TM) reconhece múltiplos pontos de interação simultâneos, frequentemente incluindo também a pressão de cada um independentemente, assim como a posição. Isso permite gestos e interação através de vários dedos ou mãos, aumentando a qualidade da utilização da tela, contribuindo para a manipulação direta através de gestos intuitivos (SCIENTIFIC AMERICAN, 2008). São diversas as abordagens de composição e estudo de TM. Estas serão tratadas na sequência de capítulos do presente texto.

A TM é uma tela que se pode ver o que aparece no monitor ou tela de algum dispositivo eletrônico e, ao tocá-la, ela responderá como se tocassem nos objetos mostrados nela. É uma tecnologia que integra não só sensores na tela, mas também programas e interfaces específicas para entender o que está acontecendo (ASSIS, 2012).

Esta tecnologia surgiu no ano de 1972 através do engenheiro eletrônico dinamarquês Bent Stumpe, utilizando de sensores de toque de capacitância e refinada pelo Conseil Européen pour la Recherche Nucléaire - CERN em 1977, para a criação da primeira interface de controle do acelerador de partícula Super Proton Synchrotron (STUMPE, 1978; BUXON, 2013).

1.1 Justificativa do tema

A proposta da realização deste trabalho monográfico de conclusão de curso é apresentar um comparativo entre as três tecnologias existentes no mercado, realizar um levantamento bibliográfico sobre as mesmas e moldar um aplicativo CIR.

Recentemente, o alcance dessa tecnologia de entrada de dados tem atingido novos níveis, impulsionados por dispositivos móveis com apelo multimídia. A motivação deste trabalho é fruto do potencial desta tecnologia e suas aplicações, considerando fundamentalmente necessidades e possibilidades de estudos e contribuições no âmbito de hardware para usabilidade de dispositivos TM (ASSIS, 2012).

1.2 Objetivos da pesquisa

1.2.1 Geral

Realizar um comparativo entre os equipamentos da PQ Labs™, Microsoft®Surface 1.0 e Microsoft®PixelSense.

1.2.2 Específico

a) Levantamento bibliográfico das três tecnologias;
b) Detalhamento dos parâmetros analisados;
c) Histórico de cada equipamento;
d) Comparativo entre as tecnologias;
e) Modelo de um aplicativo Collaborative Information Retrieval (CIR) voltado para equipamentos TM;
f) Levantamento de questões para novos estudos.

1.3 Organização do trabalho

Esta monografia encontra-se organizada em capítulos. Demonstrou-se no capítulo 1 a introdução necessária ao restante dos capítulos que compõem o trabalho.

No capítulo 2, mostra-se a fundamentação teórica da pesquisa. De início, apresentam-se os históricos e a tecnologia das TM com ênfase nos equipamentos PQ Labs™, Microsoft®Surface 1.0 e Microsoft®PixelSense. As subdivisões do Gerenciamento de Configuração de *Software* encontram-se dispostas em tópicos dentro do capítulo.

Revelam-se no capítulo 3, os tipos de TM: Capacitiva, Resistiva, Infravermelho (IR) e Surface Acoustic Wave. Estas serão abordadas nos três tipos de equipamentos citados acima, fazendo-se um comparativo entre os mesmos nos quesitos quantidade de toques simultâneos, tamanho de tela, softwares requeridos e custo.

No capítulo 4, será abordado o ambiente de desenvolvimento Software Development Kit (SDK) entre os três equipamentos mencionados anteriormente, realizando um comparativo através de tabela com os itens: linguagens de programação, ferramentas, sistemas operacionais, tamanho do SDK e facilidade de implementação.

Uma proposta de cenário de uso será mostrada no capítulo 5. Um aplicativo CIR será moldado para demonstração do uso da tecnologia da PQ Labs™, por obedecer ao conceito CIR, pela implementação e pelo baixo custo, mostrando a interação dos objetos reais e virtuais através de toque simultâneos em um universo multiusuário.

Por fim, apresentam-se no Capítulo 6 as considerações finais e o futuro referente a esta pesquisa.

2 TECNOLOGIA DE TELAS DE MULTITOQUE

As TM ganharam força na década de 80, mais precisamente em 1982, na Universidade de Toronto, com a criação da primeira TM de uso humano. Consistia em um vidro fosco associado a uma câmera e, ao pressionar o vidro com um ou vários toques, a câmera era capaz de identificá-los através de pontos escuros em um fundo branco. Através da pressão, era identificado o tamanho dos pontos, tornando-se também uma tela sensível à pressão (JIALIANG *et. al.*, 2012).

Várias outras empresas também contribuíram para o crescimento da tecnologia de TM. Podem-se destacar a Fingerworks™ (adquirida pela Apple™ em 2005), Microsoft®, Mitsubishi Electric® e PQ Labs™.

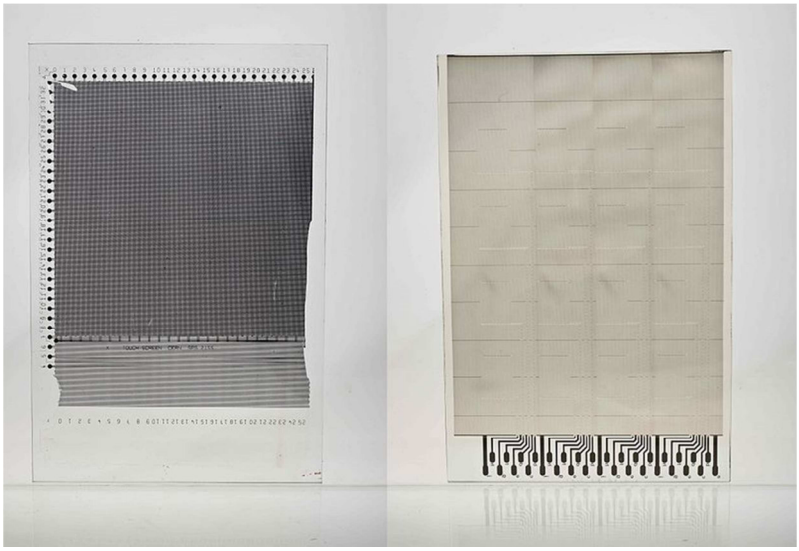


Figura 1 – Protótipo CERN-STUMPE de sensor de capacitância
Fonte: STUMPE, 1978

2.1 PQ Labs™

A PQ Labs™ é uma empresa norte americana, localizada no Vale do Silício, San Jose - Califórnia. Fundada em 2008, é pioneira no desenvolvimento de TM do tipo overlay (PQ LABS ©, 2013).

As telas do tipo overlay se caracterizam em sobrepor uma TM em um monitor para o controle das funções do equipamento em questão. Se o equipamento for um computador, hipoteticamente, rodando o sistema operacional Microsoft®Windows, ao colocar-se a tela da PQ Labs™ em frente a este monitor e

fazer as instalações corretas, o computador será controlado através da TM, onde a interação será através da tela e não mais com os periféricos comuns de entrada, como o teclado e o mouse (RIBEIRO, 2013).

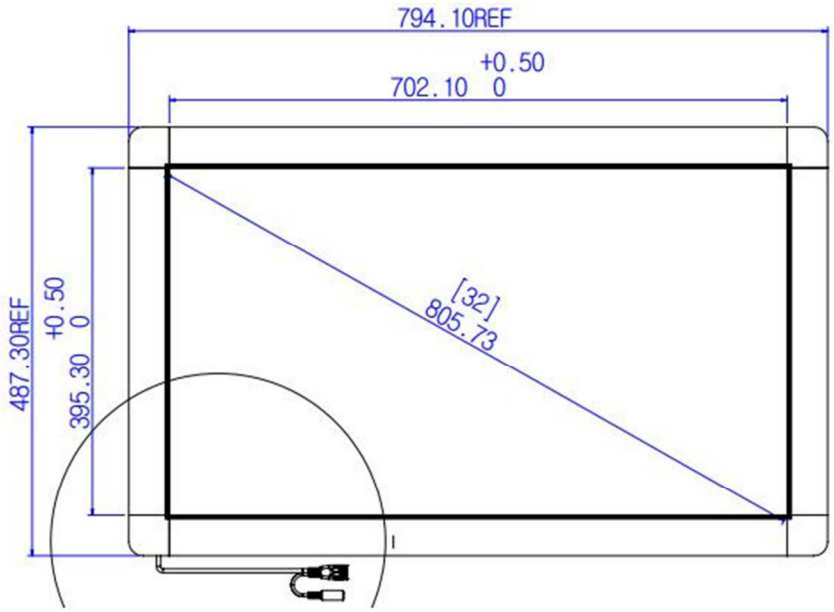


Figura 2 – Tela Overlay de 32”
Fonte: http://multi-touch-screen.com/

A tecnologia utilizada nas telas da PQ Labs™ é a do tipo IR. A alimentação é feita por duas portas USB, o que significa uma tensão de 5V. Toda a estrutura da tela é confeccionada em alumínio e a espessura do vidro temperado é de 3 mm (KUMPARAK, 2012).

2.2 Microsoft®Surface 1.0

O Microsoft®Surface 1.0 foi o primeiro projeto de TM da empresa norte-americana Microsoft®, sediada em Redmond – Washington. (WALKER, 2012).

Ainda segundo Walker (2012), o equipamento foi idealizado no ano de 2001 e no ano de 2003, o mesmo foi apresentado ao CEO da Microsoft®, Bill Gates, que deu aval para o prosseguimento do mesmo. Em 2005 foi apresentado o primeiro protótipo, sendo comercialmente lançado no ano de 2008.

Trata-se de uma mesa interativa que, na sua concepção de reconhecimento de toques, emite luz infravermelha na tela. Um objeto colocado

sobre a tela refletirá a luz infravermelha projetada na superfície. A luz refletida é capturada por quatro câmeras infravermelhas, que estão colocadas dentro do equipamento, conforme demonstrado na figura 3 (GONÇALVES; TOMÁS, 2012).

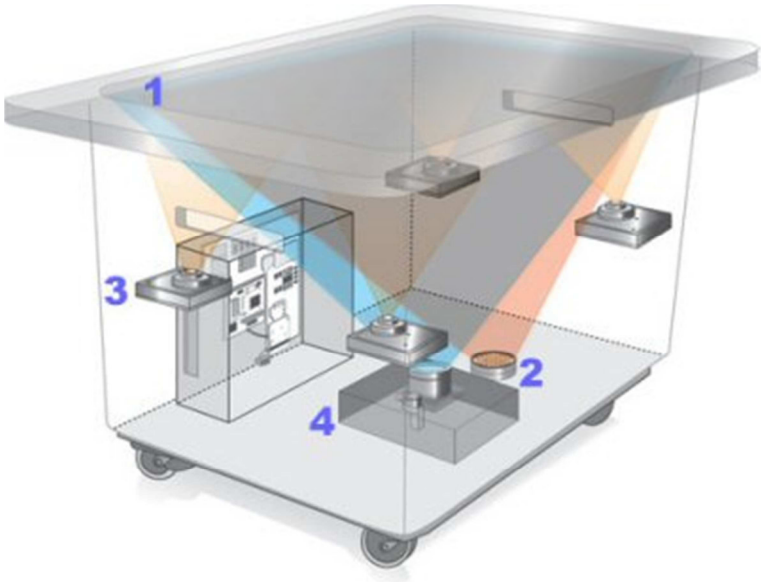


Figura 3 – Microsoft®Surface 1.0: Diagrama. *1 = tela, 2 = emissor de luz infravermelha, 3 = câmera infravermelha e 4 = projetor.

Fonte: SOETENS, 2012.

A definição das localizações dos toques é feita em três passos. Primeiro, os pontos luminosos (*blobs*) são definidos aplicando um filtro da imagem capturada. Os locais são escolhidos através de uma imagem em preto e branco, onde contém pixels brancos nas áreas dos *blobs*. Qualquer distorção luminosa do ambiente pode ser filtrada aplicando diversas técnicas de renderização na imagem em preto e branco (SOETENS 2012).

Soetens (2012) ainda afirma que o segundo passo consiste em agrupar as regiões dos pixels definidos na imagem em preto e branco. Cada região representa um toque individual. A partir dessas regiões, o ponto exato de toque pode ser deduzido. O ponto de toque é definido como o centro da região. Um algoritmo de rastreamento é usado para monitorar a localização do ponto de toque sobre uma série de frames subsequentes. Às vezes, dois dedos adjacentes na superfície da tela podem representar uma única região de pontos. Pelo tamanho da área de *blobs* e pela informação dos frames anteriores do algoritmo de rastreamento, conclui-se que aquela área dos *blobs* foi gerada por múltiplos contatos na tela. Nesse caso, vários toques simultâneos são reconhecidos naquela região.

O último passo é mapear os toques processados da imagem diretamente no sistema de coordenadas da tela, aplicando uma transformação homográfica nos pontos (CHABAN; AKERBERG, 2012).

Nesta primeira versão, para posteriormente ser renomeado como Microsoft®PixelSense e utilizar uma nova tecnologia de TM, o hardware ficava a cargo da Intel, fornecendo a *Central Processing Unit* (CPU), e da ATI, responsável pela *Graphics Processor Unit* (GPU). A tela possuía a resolução *High Definition* (HD), ou seja, até 720 linhas e com tamanho fixo de 30 polegadas. Faz-se presente o bluetooth, portas *Universal Serial Bus* (USB), leitor de cartão, rede *Local Area Network* (LAN) e rede Wi-Fi. O sistema operacional do equipamento, responsável pelo seu funcionamento, foi o Microsoft Windows Vista (MICROSOFT SURFACE©, 2013).

A figura 4 revela o equipamento em sua 1ª versão.

Figura 4 – Microsoft®Surface 1.0
Fonte: MICROSOFT SURFACE©, 2013.

Ao final de 2011, o hardware sofreu um upgrade. A tela, atualmente, é fornecida pela Samsung, modelo SUR40. Com isso, mudou-se o nome comercial, passando a se chamar Microsoft®PixelSense. O nome Surface foi realocado para a linha de tablets da empresa em 2012 (SOETENS, 2012).

2.3 Microsoft®PixelSense

Essa é a 2ª versão do Microsoft Surface 1.0. Foi apresentado na *Consumer Electronics Shows* (CES) do ano de 2011 e disponibilizado para o mercado no começo de 2012. Foram feitas mudanças significativas de hardware e de conceito da 1ª para a 2ª versão (MICROSOFT® SURFACE® 2.0, 2012).

As quatro câmeras infravermelhas foram substituídas por uma série de pequenos sensores IR, localizados em cada pixel da tela do Microsoft®PixelSense 2.0, demonstrado na figura 5 (SOETENS, 2012).

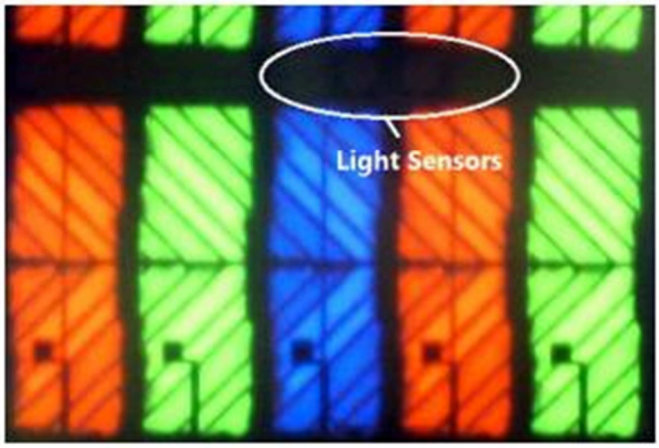


Figura 5 – Sensores IR Microsoft®PixelSense
Fonte: SOETENS, 2012

Nesse caso, as coordenadas de sistema da imagem são iguais às coordenadas da tela, e isso significa que a transformação homográfica não é mais necessária. Em oposição à tecnologia capacitiva de TM, a tela não precisa de contato condutivo. Entretanto, a técnica de redução de ruído na imagem tem que ser aplicada diretamente na imagem infravermelha, sendo assim, pequenos *blobs* não serão processados pelo algoritmo de rastreamento (CHABAN; AKERBERG, 2012).

Em termos físicos, isso implicou em uma diminuição no tamanho e no peso do equipamento, em virtude da tecnologia de sensores IR estarem alocados na própria tela. Essa mudança permitiu que o equipamento pudesse ser montado tanto na horizontal como na vertical, melhorando assim a sua usabilidade (WALKER, 2012).

Ainda segundo Walker (2012) a tela é fornecida pela empresa coreana Samsung™, modelo SUR40, com resolução FullHD (1920x1080 pixels) e 40

polegadas de tamanho. Em termos de hardware, o equipamento também sofreu uma série de mudanças. A empresa fornecedora da CPU passou a ser a AMD. A GPU continuou sendo fornecido pela ATI, empresa subsidiária da AMD.

Figura 6 – Microsoft®PixelSense
Fonte: MICROSOFT® PIXELSENSE®, 2013

As conexões de entrada e saída foram preservadas, porém sofrendo melhorias e adições de uma saída *High-Definition Multimedia Interface* (HDMI), interface de LAN Gigabit e saída ótica digital com canais 5.1. No que se refere ao sistema operacional, o Windows® Vista utilizado na versão Microsoft®Surface 1.0 foi substituído pelo Windows® 7 Professional Embedded Edition 64bits (MICROSOFT® PIXELSENSE®, 2013).

3 HARDWARE MULTITOQUE

TM não são exatamente tecnologias novas. Várias formas de implementação foram surgindo durante a década de 70. Múltiplas patentes demonstram como superfícies sensíveis ao toque, baseadas em câmeras e sensores, podem ser construídas (PETER *et. al.*, 2008).

O multitoque é uma tecnologia de interação que se baseia na detecção o de um ou mais contatos sobre uma superfície. A grande vantagem do multitoque em relação a outras tecnologias, como o toque simples ou mouse, é permitir a utilização

dos nossos membros de uma forma mais eficiente e natural, tornando o mundo virtual mais próximo do real (SILVA, 2011).

Pinto (2011) relata que através do conjunto de suporte necessário formado pelos circuitos eletrônicos e partes eletromecânicas, esta tecnologia se torna possível. Vários hardwares de multitoque estão disponíveis no mercado, porém o funcionamento geral desse tipo de soluções não foge muito a regra conforme demonstrado na figura 7.

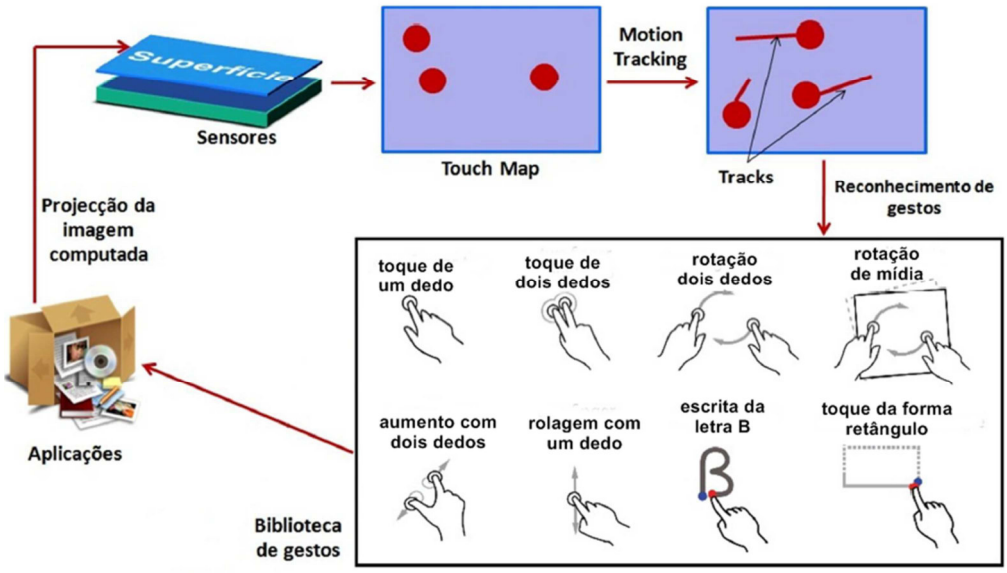


Figura 7 – Arquitetura genérica de um sistema multitoque.

Fonte: PINTO, 2011. Tradução nossa.

A superfície interativa capta os *blobs* para formar um mapa de toques. O processo de monitoramento trata de “seguir” o deslocamento dos *blobs*, de forma a obter um conjunto de movimentos. Essa informação é depois comparada com uma biblioteca de gestos previamente armazenados. Caso seja detectado um gesto conhecido, é gerado um evento que será integrado na aplicação. Essa informação é processada, dando origem à imagem projetada na superfície (GESTUREWORKS, 2010).

Existem duas técnicas para captar toques em uma superfície, que são:

a) **Tecnologia Elétrica:** São mais difíceis de implementar. Os materiais não são tão comuns como os necessários para técnicas de tecnologia ótica, mas por outro lado, são mais utilizados em equipamentos de pequenas dimensões porque permitem montagens mais compactas (PINTO, 2011).

b) **Tecnologia Ótica:** Relativamente fáceis de implementar, pois basta utilizar materiais relativamente acessíveis, como um projetor para projetar a imagem na superfície, uma tela para “reter” essa projeção e uma câmera que capte a luz infravermelha.

Com base nessas técnicas, foram estudados os tipos de telas que utilizam as seguintes tecnologias:

a) Resistiva (elétrica);
b) Capacitiva (elétrica);
c) IR (ótica);
d) Surface Acoustic Wave (ótica).

3.1 Tecnologia Resistiva

As telas com esta tecnologia utilizam um conjunto composto por um painel de vidro ou acrílico e por duas camadas condutivas de metal, separadas por um pequeno espaço entre estas camadas, ilustradas na figura 8. Quando a tela está ligada, corre uma corrente elétrica através dessas duas camadas. O toque de um dedo na superfície exterior origina o contato entre as duas camadas condutoras, com uma consequente descarga elétrica entre elas. Esta descarga provoca uma divisão de tensão entre os dois condutores nesse ponto, obtendo-se as coordenadas para o mapa de toques (NCR CORPORATION, 2007).

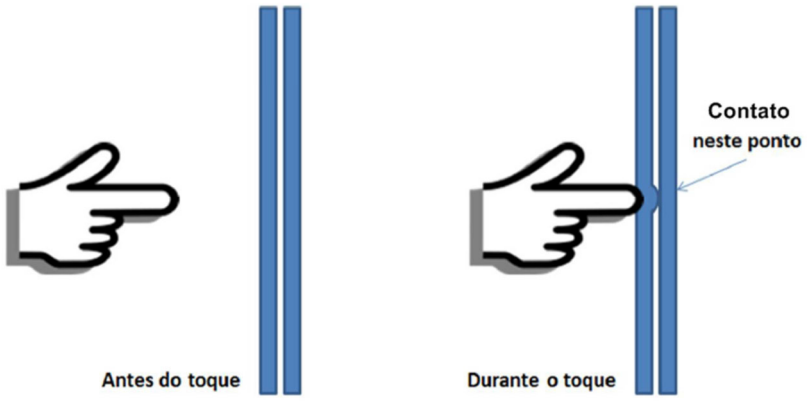


Figura 8 – Funcionamento de uma superfície com tecnologia resistiva.
Fonte: PINTO, 2011. Tradução nossa.

Através do conceito de divisor de tensão, pode-se saber a sua posição em coordenadas (x,y). Uma tela resistiva oferece no máximo 85% de precisão e sempre necessitam ser calibradas por causa do desgaste do material, o que acaba mudando a tensão de referência. São as telas mais baratas do mercado (ASSIS, 2012).

Ainda de acordo com Assis (2012) esta tecnologia é a mais comum, mas possui duas desvantagens. A primeira é que por serem compostas por várias camadas, elas obstruem mais a passagem da luz, oferecendo no máximo 85% de transparência, tornando a tela levemente opaca, além de reduzirem o ângulo de visão. O segundo problema é que elas são pouco sensíveis a toques diretos com os dedos (ASSIS, 2012).

3.2 Tecnologia Capacitiva

Telas capacitivas possuem tecnologia diferente: o conceito é baseado em capacitores (feitos por um óxido fraco), os quais armazenam energia e descarregam de acordo com seu uso. Os chamados “sensores capacitivos” são envolvidos por uma retícula de fios, dispostos entre duas camadas de vidro. É alimentado com uma pequena tensão, de modo a acumular energia. O corpo humano também é um capacitor, e, quando em contato com a superfície, retira alguns elétrons da mesma, detectando a posição do toque. Ao funcionar com troca de elétrons, a superfície não detecta toques de canetas de pontas arredondadas, somente dos dedos ou de canetas especiais. Um contraste relevante em relação à tela resistiva, onde qualquer objeto é capaz de fazer as duas camadas condutoras se tocarem (SCHÖNING, 2008).

Como é representado na figura 9, no painel de vidro é colocada uma camada que armazena uma carga elétrica. Quando um dedo entra em contato com a tela, no ponto desse toque, é originada uma pequena descarga elétrica na camada capacitiva, que é calculada pela diferença relativa de carga, e dessa forma são calculadas as coordenadas (x,y) do ponto de contato (HAJAD; ASMA, 2008).

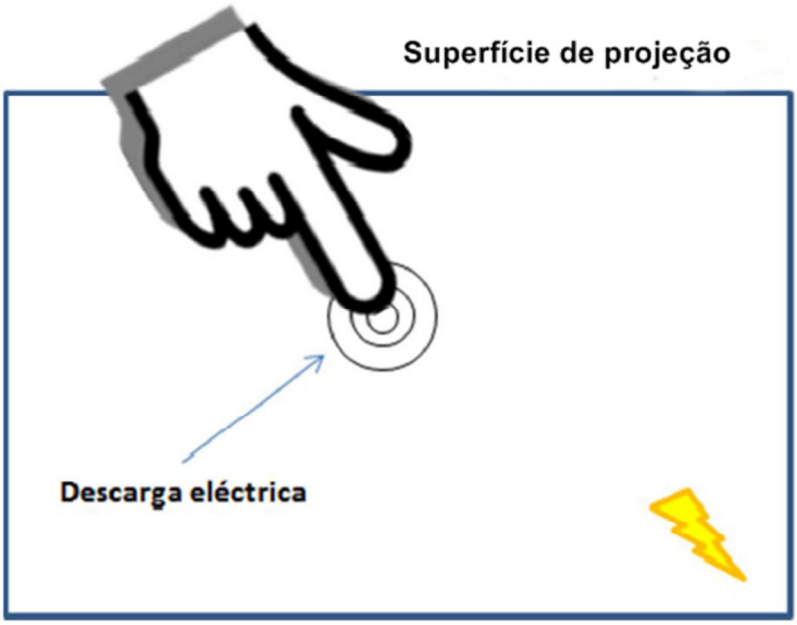


Figura 9 – Funcionamento de uma superfície com tecnologia capacitiva.
Fonte: PINTO, 2011.

A precisão deste tipo de tela é praticamente 100%, tem alta durabilidade, mas não pode trabalhar em ambientes muito frios ou quentes, além do fato de ter a leitura prejudicada pela poeira ou marcas de dedos. Uma vantagem desta tecnologia relativamente à tecnologia resistiva é que esta permite a passagem de cerca de 90% da luz emitida pela projeção. Portanto, a perda de luminosidade original da projeção é maior em uma superfície com tecnologia resistiva do que uma projeção em uma superfície com tecnologia capacitiva e isto acontece porque na tecnologia resistiva são utilizadas mais camadas para a captação dos toques. Outra vantagem desta tecnologia relativamente à tecnologia resistiva é a sua maior durabilidade (PINTO, 2011).

3.3 Tecnologia IR

De uma forma geral, as TM de funcionamento ótico têm por base a utilização de um emissor de luz infravermelha (que não é visualizado pelo olho humano), uma câmera modificada de forma a visualizar a luz infravermelha e um projetor ou tela de cristal líquido, diferindo apenas na forma como são colocados estes elementos. As principais técnicas óticas para as TM são:

a) FTIR (*Frustrated Total Internal Reflection*);
b) DI (*Diffused Ilumination*);
c) LLP (*Laser Light Plane*);
d) DSI (*Diffused Surface Illumination*).

3.3.1 FTIR (Frustrated Total Internal Reflection)

São colocados *diodos de emissores de luz* (LED) IR nas extremidades da tela que se vão espelhar no painel acrílico e, quando este é pressionado, cria-se um *blob* que é captado pela câmera de IR. Esta técnica baseia-se num fenômeno ótico, que ocorre quando um raio de luz atravessa um material vindo de outro material, com um índice de refração superior e um ângulo de incidência maior. Nestas condições, não há refração desse raio, e a luz é totalmente refletida dentro da superfície, ficando “aprisionada” (SOUZA, 2010).

O *blob* é captado por uma câmera de IR, colocada debaixo da superfície (ALIKUTTY, 2008). Na figura 10 pode-se observar a reflexão interna, representada por linhas vermelhas.

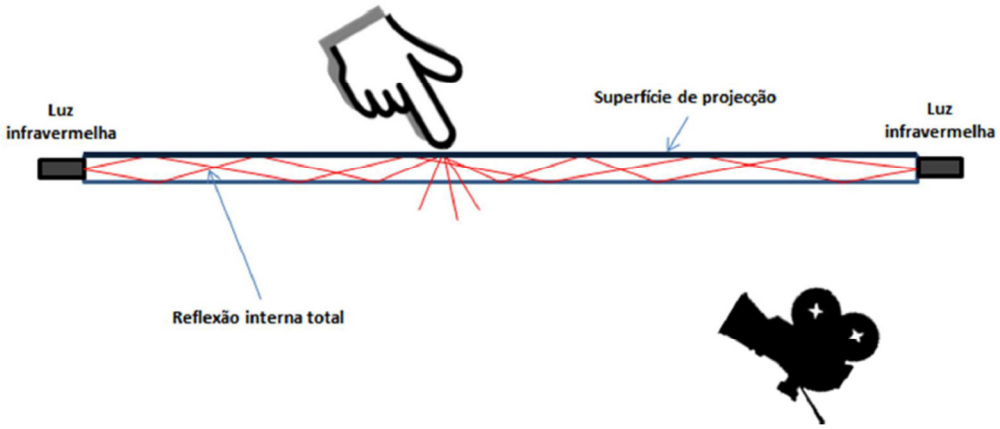


Figura 10 – Funcionamento da técnica FTIR
Fonte: PINTO, 2011

3.3.2 DI (Diffused Ilumination)

São projetadas luzes de IR para o acrílico e quando este é pressionado, cria-se um *blob* captado pela câmera de IR (COSTA, 2010). A figura 11 demonstra o funcionamento.

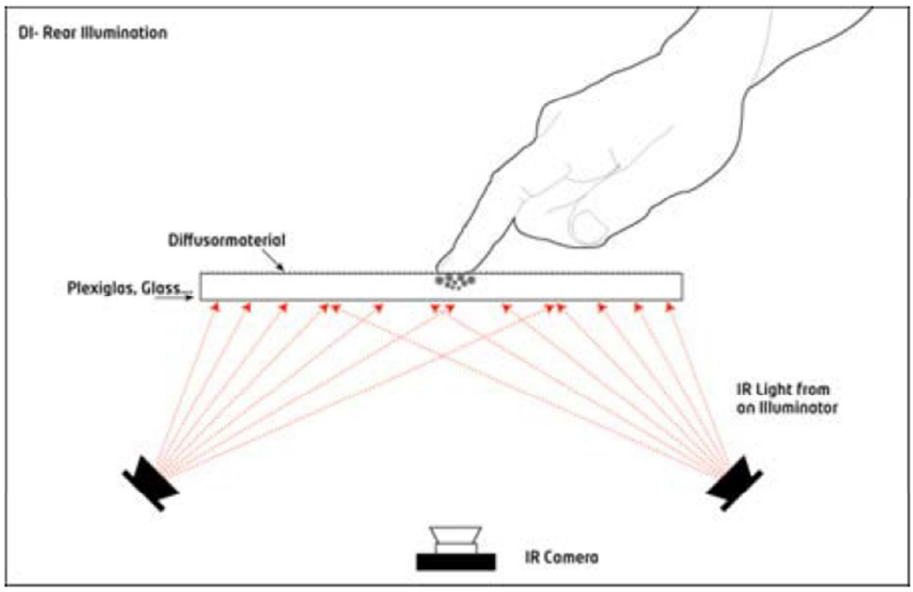


Figura 11 – Funcionamento da técnica DI
Fonte: COSTA, 2010

3.3.3 LLP (Laser Light Plane)

De acordo com Silva (2011), utilizam-se emissores de raios lasers IR que se encontram na borda do equipamento, para criar um plano laser ligeiramente acima da tela e, ao se pressionar, o *blob* criado é captado pela câmera de IR, como demonstrado na figura 12.

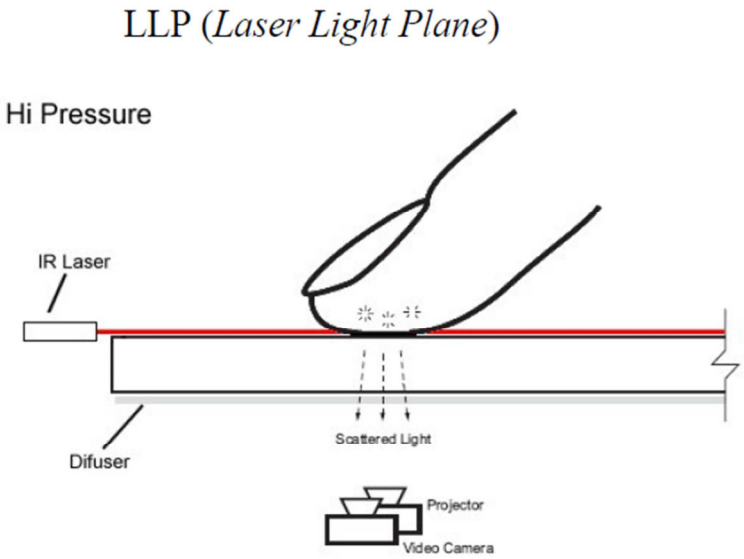


Figura 12 – Funcionamento da técnica LLP
Fonte: COSTA, 2010

3.3.4 DSI (Diffused Surface Illumination)

São colocados LED IR nas extremidades da tela que espelharão na superfície e, ao ser pressionada, um blob é criado, sendo captado pela câmera de IR, de acordo com a figura 13 (SCHÖNING, 2008).

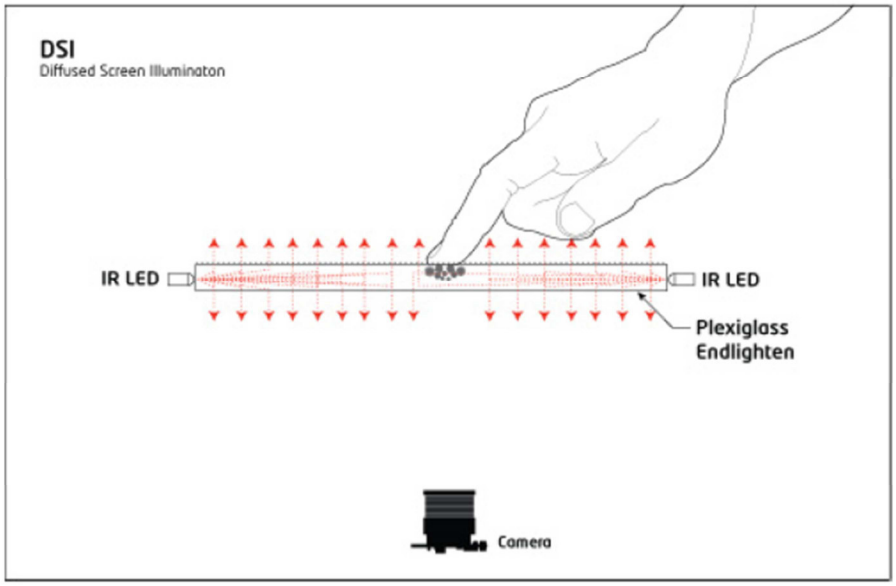


Figura 13 – Funcionamento da técnica DSI
Fonte: SCHÖNING, 2008

3.4 Surface Acoustic Wave

Sistemas que utilizam esta tecnologia são similares aos que usam a tecnologia IR. Os transdutores transmissores e receptores, tanto para o eixo x como o y, são montados na superfície e ondas ultrassônicas (*waves*) são criadas na tela de vidro e direcionadas por refletores. Ao processar as *waves* em sinais eletrônicos e observando as mudanças quando a superfície é tocada, é possível calcular a posição do toque (SCHÖNING, 2008).

A tecnologia permite passar 100% da luminosidade da tela, pois, diferentemente das outras tecnologias, ela não posiciona nenhuma placa metálica sobre esta (ASSIS, 2012).

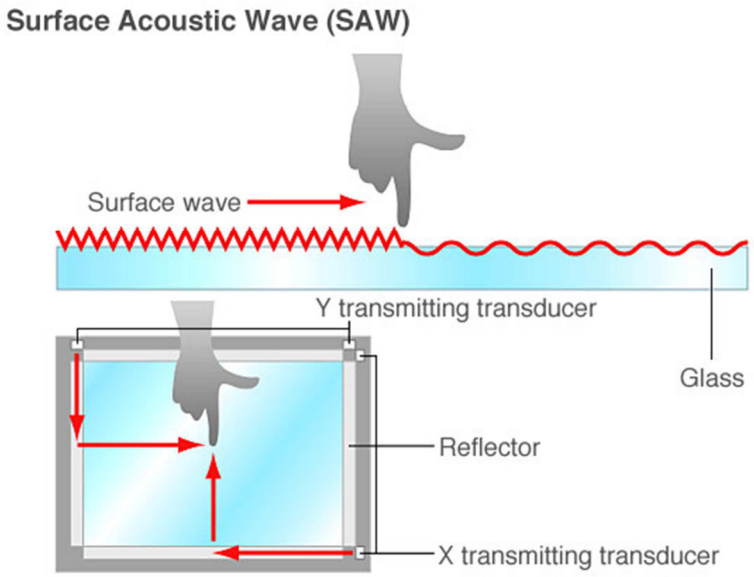


Figura 14 – Tecnologia Surface Acoustic Wave
Fonte: EIZO®, 2012

3.5 Comparativo entre as tecnologias

Discriminadas as tecnologias de hardware de TM, a tabela abaixo mostra o comparativo entre as telas da PQ Labs™, Microsoft®Surface 1.0 e Microsoft®PixelSense quanto ao tipo de tecnologia utilizada, custo, quantidade de toques simultâneos, dimensões, tamanho de tela e o sistema operacional compatível.

Tabela 1 – Comparativo entre as tecnologias de telas multitoques
Fonte: do Autor

C / E	PQ Labs	Microsoft®Surface 1.0	Microsoft®PixelSense
Tipo de Tela	IR (Diffused Surface Illumination)	Capacitiva / IR (Diffused Illumination)	Capacitiva / IR (Light Sensors)
Custo	Médio	Alto	Alto
Qtd. de Toques	32 toques	32 toques	50 toques
Dimensões	Grande e fino	Pequeno e espesso	Médio e fino
Tamanho de Tela	Variável (21,5” a 55”)	Fixo (30”)	Fixo (40”)
Sis. Operacional	Windows® 8/7/Vista/XP, Apple™ OS X, Linux	Windows® Vista	Windows® 7 Embedded

4 AMBIENTE DE DESENVOLVIMENTO SDK

Em virtude da tecnologia multitoque não ser padronizada, os eventos gerados são específicos de cada tecnologia. Sendo assim, frameworks de cada hardware foram criados para reconhecimento de gestos e eventos e desenvolvimentos de aplicações. Como resultado, desenvolvedores precisam utilizar as soluções de SDK para cada tipo de equipamento e seu hardware específico (KHANDKAR *et. al.*, 2010).

SDK é um conjunto especial de ferramentas de desenvolvimento específicas para a criação de aplicações a uma plataforma. Essa plataforma pode ser de vários tipos como hardware, sistema computacional, estações de jogos, sistemas operacionais ou até mesmo outro software.

O SDK sempre é fornecido pela empresa desenvolvedora da plataforma. São distribuídos na página das empresas e gratuitamente a fim de encorajar o uso da plataforma e/ou linguagem.

4.1 PQ Labs™ SDK

O PQ Labs™ possui uma linha variada de SDK para Microsoft® Windows® e Apple™ OS X. No Windows®, há três opções de se escrever aplicativos: através do SDK e biblioteca de gestos PQ Labs™, da biblioteca Windows Presentation Foundation (WPF) e através do protocolo Tangible User Interface Object (TUIO). No Apple™ OS X há duas possiblidades: SDK e biblioteca de gestos PQ Labs™ e protocolo TUIO (PQLABS©, 2013).

As tabelas 2 e 3 informam quais são as linguagens utilizadas em cada plataforma e suas opções. O desenvolvedor escolherá aquela que mais se adequa ao tipo de trabalho proposto, disponibilidade de linguagem de programação e sistema operacional.

Tabela 2 – Informativo da plataforma Windows® e opções de linguagens
Fonte: PQ Labs©, 2013

Microsoft® Windows®		
Opções	**Linguagens de Programação**	**Sistemas Operacionais**
SDK e biblioteca de gestos PQ Labs™	C/C++, C# (WPF), C# (DLL Import), Flash/Flex/AIR AS3, Flash AS2, Java	Windows® 8/7/Vista/XP
WPF	C/C++, C# (WPF), Flash (versão 10.1 em diante), AIR 2.0	Windows® 7 (não aplicável as versões Home Basic e Stater Edition)
Protocolo TUIO	C/C++, Flash (AS3), Java, C# (WPF)	Windows® 8/7/Vista/XP

Tabela 3 – Informativo da plataforma Apple™ OS X e opções de linguagens
Fonte: PQ Labs©, 2013

Apple™ OS X		
Opções	**Linguagens de Programação**	**Sistemas Operacionais**
SDK e biblioteca de gestos PQ Labs™	C/C++, Objective-C, Flash/Flex/AIR, Java	Apple™ OS X (sem restrição de versão)
Protocolo TUIO	C/C++, Flash (AS3), Java	Apple™ OS X (sem restrição de versão)

O SDK consiste em arquivos base para desenvolvimento, acompanhados de um guia de referências e/ou um arquivo de ajuda. Porém, sempre se faz necessário a presença de um programa de codificação e compilação do aplicativo (WESTING, 2011).

As tabelas 4 e 5 informam quais programas são recomendados para cada tipo de pacote, o tamanho em disco do SDK e formato de compactação oferecido, tanto para Windows como para o Apple™ OS X.

Tabela 4 – Detalhes dos SDK para desenvolvimento no Windows®

Fonte: PQ Labs©, 2013

Microsoft® Windows®			
Pacote	**Programa Requerido**	**Tamanho em Disco**	**Compactação**
C/C++	Microsoft® Visual Studio	2,65MB	.ZIP
C# (WPF)	Microsoft® Visual Studio	9,92MB	.EXE
C# (DLL Import)	Microsoft® Visual Studio	2,34MB	.ZIP
Flash/Flex/AIR AS3	Adobe® Flash CS Series, Flex Builder, Flash Builder	3,57MB	.EXE
Flash AS2	Adobe® Flash CS Series, Flex Builder, Flash Builder	769KB	.ZIP
Java	Eclipse, MyEclipse	2,54MB	.ZIP

Tabela 5 – Detalhes dos SDK para desenvolvimento no Apple™ OS X

Fonte: PQ Labs©, 2013

Apple™ OS X			
Pacote	**Programa Requerido**	**Tamanho em Disco**	**Compactação**
C/C++	XCode	2,65MB	.ZIP
Objective-C	XCode	3,99MB	.ZIP
Flash/Flex/AIR	Adobe Flash CS Series, Flex Builder 3, Flash Builder 4	2,45MB	.ZIP
Java	Eclipse, MyEclipse	1,95MB	.ZIP
Java	Eclipse, MyEclipse	2,54MB	.ZIP

A figura 15 ilustra um fluxograma clássico para desenvolver aplicativos compatíveis com o PQ Labs™.

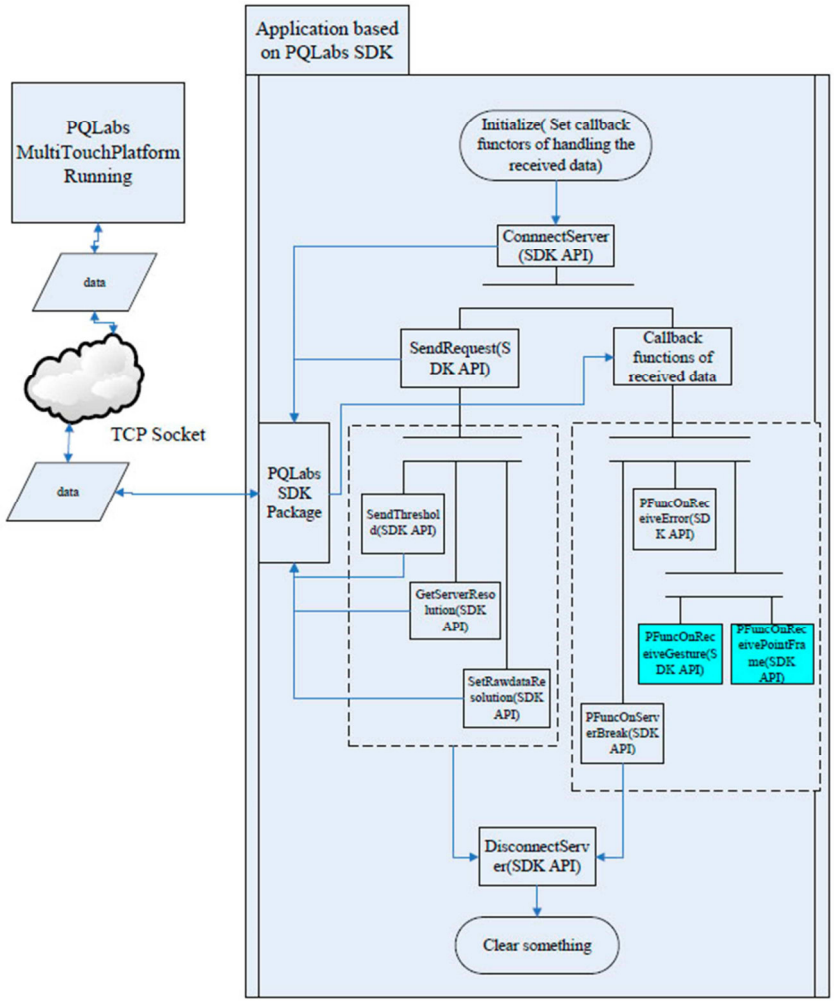


Figura 15 – Fluxograma padrão SDK PQ Labs™

Fonte: PQ Labs© Multi-Touch SDK Reference V1.3.7

A tabela 6 lista todas as classes e Application Programming Interface (API) utilizadas pelo SDK PQ Labs™. Ela é comum para qualquer tipo de implementação, obedecendo ao padrão WPF.

Tabela 6 – Lista de classes e API do SDK PQ Labs

Fonte: PQ Labs© Multi-Touch Screen G2 SDK Reference V1.2. Tradução do Autor

Classes	
Nome	**Descrição**
BaseGesture	BaseGesture é a classe base para todos os gestos de toque e manipuladores
BaseGestureEventArgs	Classe base para GestureEventArgs, RotateEventArgs, ZoomEventArgs, DragMoveEventArgs
BaseTouchEventArgs	Classe base para todo o toque e gesto relacionados eventos roteados

BlobInfo	Classe que armazena as informações de toque local em um certo controle, como touchID, posição, tamanho, etc.
DragMoveEventArgs	Contém dados de eventos associados a um DragMoveEvent.
DragMoveGesture	Calcula o centro de todas as bolhas de toque com DefaultInterpretor, e tomá-lo como um único
DragMoveEvent	Um exemplo MobiGesture único é adicionado ao Gesto DragMove atual quando construído.
DragScaleRotate	A classe pode aplicar o gesto de toque sobre o UIElement facilmente, ele contém manipulador gesto. O recipiente do UIElement deve ser uma tela. Ele suporta não mais de 2 pontos para arrastar, redimensionar e rodar o UIElement.
DSRManipulator	Uma classe abstrata herdada de Manipulator. inicializa o vetor usado no Manipulator.
DSRProcessor	Processador Drag Scale Rotate
DSRSmoothor	Drag Scale Rotate smoothor
DualClickGesture	Enviará um DualClickEvent quando há DualDown e DualUp ocorreu perto o suficiente no tempo e distância
DualDownGesture	Enviará um DualDownEvent quando DualDownGesture enviará evento GestureEvent.DUAL_DOWN quando a primeira para baixo e segundo para baixo ocorreu perto o suficiente no tempo
DualMoveGesture	Enviará um DualMoveEvent quando há apenas dois blobs de toque no controle e ambos estão em movimento
DualTouchGesture	Enviará um evento Touch Dual, quando dois toques na UIElement
DualUpGesture	Enviará um DualUpEvent quando há apenas dois blobs de toque em um controle, e ambos se perto o suficiente no tempo.
DurativeGesture	Classe base para gestos que podem ser levantadas com a devida GestureState (GestureState.Begin, GestureState.Progress, GestureState.End)
EventGesture	Classe base para toda a classe gestos que despacha apenas um evento certo gesto para uma classe
FlickEventArgs	Definição de flickeventargs
FlickGesture	Enviará um MultiMoveEvent quando há, pelo menos, três blobs de toque no controle
GestureEventArgs	Contém dados de eventos associados a um evento de gesto
ManipulateEventArgs	Herdado de base GestureEventArgs, contém os parâmetros em manipular gestos
ManipulateGesture	Enviará um MultiMoveEvent quando há, pelo menos, três blobs no controle
ManipulationProcessor	Manipulator Processor, ela é herdada DSRProcessor

Manipulator	Inicialização do Manipulator
MultiDownGesture	Enviará um MultiDownEvent quando há outro toque para baixo de ações ocorreu perto o suficiente depois de uma Ação DualDown
MultiDragScaleRotate	A classe pode aplicar o gesto de toque sobre o UIElement facilmente, ele contém manipulador gesto. O recipiente do UIElement deve ser uma tela. Ele invoca um método de arrastar, redimensionar e rodar o UIElement
MultiMoveGesture	Enviará um MultiMoveEvent quando há, pelo menos, três blobs de toque no controle
MultiTouch	Classe contém controles globais para funções multi-touch específicas
MultiUpGesture	Enviará MultiUpEvent quando todos os blobs de toque (pelo menos três) sobre o controle são em um curto espaço de tempo
OnePauseOneMoveGesture	Enviará OnePauseOneMoveEvent quando há dois blobs de toque no controle, um deles está se movendo enquanto o outro não está se movendo
Processor abstract class	Processa informações sobre toque, e em seguida, declara o gesto
RotateEventArgs	Contém dados de eventos associados a um RotateEvent
RotateGesture	Calcula o ângulo de rotação de apenas dois blobs de toque com DefaultInterpretor, e tomá-lo como um RotateEvent único. Um exemplo OnePauseOneMoveGesture é adicionado RotateGesture atual quando construído
SecondClickGesture	Enviará um SecondClickEvent quando há uma SecondDownEvent e um SecondUpEvent ocorreu perto o suficiente
SecondDownGesture	Enviará SecondDownEvent, quando já existe um blob de toque no controle
SecondUpGesture	Enviará um SecondUpEvent, quando ainda há um blob de toque no controle após este contato
SingleClickGesture	Enviará um SingleClickEvent quando há uma SingleDownEvent e um SingleUpEvent ocorreu perto o suficiente
SingleDownGesture	Enviará SingleDownEvent, se este é o abetos pousar no controle
SingleMoveGesture	Enviará um SingleMoveEvent quando há apenas um blob de toque no controle e ele está se movendo
SingleTouchGesture	Enviará um SingleTouchEvent quando há apenas um blob de toque no controle, não importa se está em movimento ou não
SingleUpGesture	Enviará um SingleUpEvent quando há apenas um toque bolhas no controle e é até agora.
Smoothor	Arrastar, escala e gira suavemente
TouchEndGesture	Enviará um SingleUpEvent quando há apenas um blob de toque no controle

TouchEvent	Classe estática para TouchEventBasic e TouchEventGesture
TouchEventArgs	Fornece dados básicos para toque eventos relacionados
TouchManager	Classe lida com todos os blobs de toque que são para baixo em um UIElement
TouchOptions	Classe que contém a biblioteca de configurações multitoque
TouchStartGesture	Enviará um SingleUpEvent quando há apenas um blob de toque no controle
ZoomEventArgs	Contém dados de eventos associados a um ZoomEvent
ZoomGesture	Calcula a proporção de escala de apenas dois blobs de toque com DefaultInterpretor, e toma-o como um ZoomEvent único

4.1.1 Protocolo TUIO

TUIO é um framework de código aberto que define um protocolo padrão e API – conjunto de rotinas e padrões estabelecidos por um software para a utilização das suas funcionalidades por aplicativos que não pretendem envolver-se em detalhes da implementação do software, mas apenas usar seus serviços – para superfícies multitoque tangíveis. O TUIO permite a transmissão de uma descrição abstrata de TM interativas, incluindo eventos de toque e descrição de objetos na superfície (KALTENBRUNNER *et. al.*, 2005).

A figura 16 define o protocolo TUIO e como é feita a sua utilização.

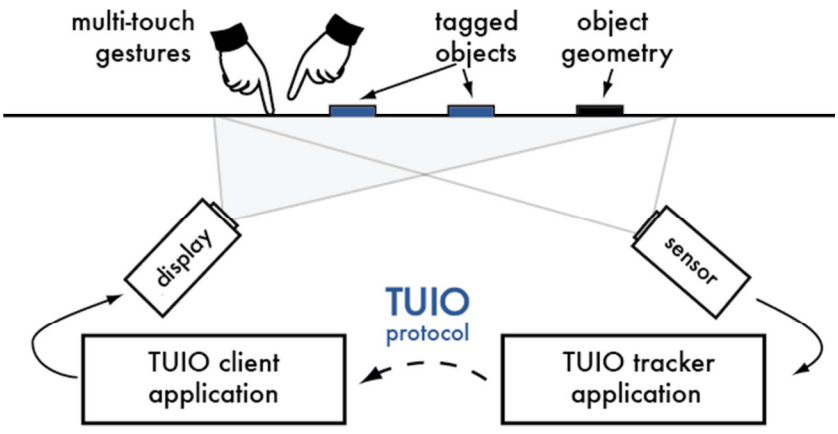


Figura 16 – Representação TUIO

Fonte: TUIO, 2013

O TUIO, assim como o WPF, são tentativas de padronização de gestos e eventos, porém, nenhum deles provê um processador de gestos em nível de

interface do usuário ou uma máquina de reconhecimento nativo (KALTENBRUNNER *et. al.*, 2005).

4.2 Microsoft®Surface 1.0 SDK

O SDK do Microsoft®Surface 1.0 cria um ambiente de simulação na estação de trabalho onde será instalado. A ferramenta Surface Simulator é responsável por gerar uma interface de usuário no computador. O SDK trabalha com o Microsoft® Visual Studio, e juntamente com os templates (modelos de documento) de projetos incluídos no SDK, permite a criação e a prototipação de aplicações multitoque na estação de trabalho sem ser necessário o uso de uma unidade do equipamento (MICROSOFT® SURFACE® SDK, 2009).

Para a instalação correta e utilização do SDK, são necessários configurações mínimas de hardware e software obrigatórios, conforme demonstram as tabelas 7 e 8.

Tabela 7 – Requerimentos de hardware para o Microsoft®Surface 1.0 SDK
Fonte: MICROSOFT® SURFACE® SDK, 2009

Monitor	Memória	Disco	Placa Gráfica
Resolução de 1280x960 ou Widescreen com resolução de 1440x900	4GB ou mais de memória RAM	4GB ou mais de espaço livre no HD	Placa compatível com Aero e 256MB ou mais de memória de vídeo

Tabela 8 – Requerimentos de software para o Microsoft®Surface 1.0 SDK
Fonte: MICROSOFT® SURFACE® SDK, 2009

Sistema Operacional	Service Pack	Ferramentas
Windows® Vista Business, Enterprise, Ultimate e Home Premium	Windows® Vista SP1	Microsoft® Visual C#® 2008 Express Edition, Microsoft® Visual Studio 2008, .NET Framework v3.5, Microsoft® XNA Framework Redistributable 2.0, Microsoft® Surface SDK 1.0 SP1, Workstation Edition e Microsoft DirectX 9.0

As figuras 17 e 18 ilustram a interface do ambiente de simulação do SDK. Vemos que há uma fidelização da TM do equipamento projetado pela Microsoft® com o que é mostrado no monitor do computador, para visualização correta das coordenadas (x,y) e das imagens projetadas.

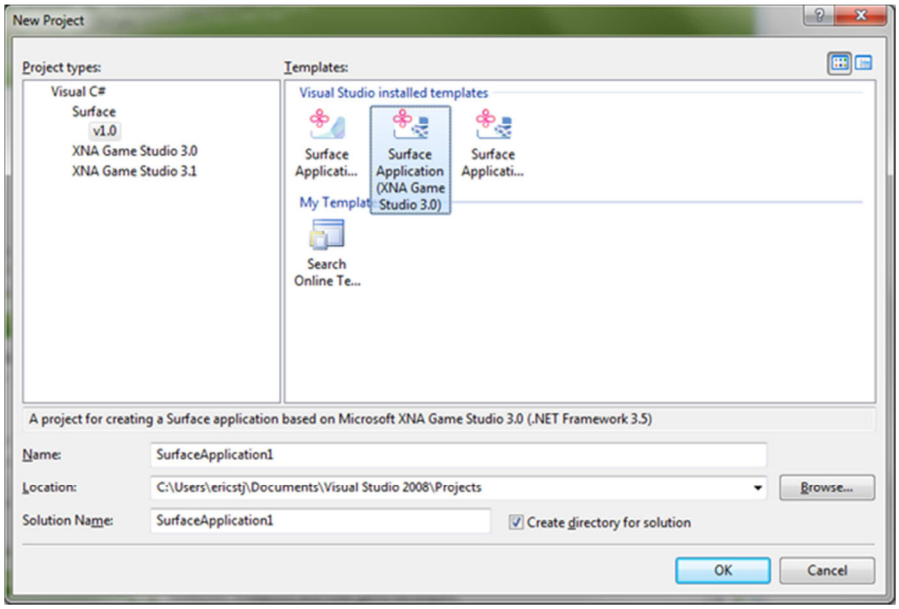


Figura 17 – Novo projeto SDK Microsoft Surface 1.0

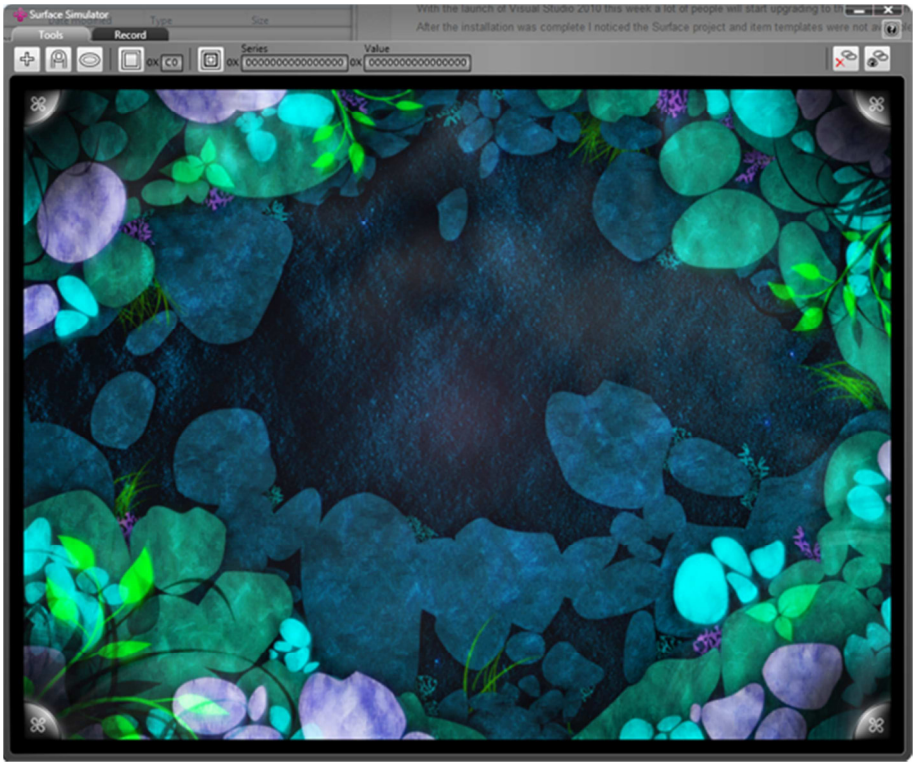


Figura 18 – Ambiente SDK Microsoft Surface 1.0

Pelos requerimentos de tantos programas adicionais para o devido funcionamento do SDK, torna-se trabalhosa a instalação correta do kit. A tabela 9 informa o tamanho do instalador do SDK, de cada programa adicional, assim como seus formatos de distribuição.

Tabela 9 – Programas requeridos para o Microsoft®Surface 1.0 SDK
Fonte: MICROSOFT® SURFACE® SDK, 2009

Programa	Tamanhos de Instalação	Compactação
Microsoft Visual Studio	607MB	.ISO
Microsoft Visual C# Express	90MB	WEBINSTALL
Microsoft XNA Framework Redistributable 2.0	2,1MB	.EXE
Microsoft Surface SDK 1.0 SP1	144MB	.EXE
Microsoft DirectX	-	Nativo no Windows Vista

4.2.1 Interação com objetos reais

Uma das inovações do Microsoft®Surface 1.0 se refere à capacidade de interação com objetos reais em sua superfície. Para que isso ocorra, precisamos classificar os objetos em dois tipos: *tagged* (marcados) e *untagged* (não marcados) (MICROSOFT® SURFACE®, 2009).

Os objetos do tipo *tagged* possuem uma etiqueta especial com um padrão de pontos visíveis ao leitor IR presente no Microsoft®Surface 1.0. Ele lê as informações contidas nessa etiqueta e toma as ações cabíveis ao qual ele foi programado, como ao ler as informações sobre uma garrafa de vinho e mostrar na tela a safra, tipo de uva, nome da vinícola e o preço do produto (MICROSOFT® SURFACE®, 2009).

Já os do tipo *untagged*, somente são reconhecidos porque tocam a superfície, mas não tem formato reconhecido e as possíveis informações que eles carregam. Para uma interação completa, é necessário que o objeto tenha uma etiqueta compatível com o Microsoft®Surface 1.0.

Em relação à sincronização de dispositivos móveis como celulares e tablets, é necessário um App (aplicativo) instalado no dispositivo móvel que interpretará, através da sua câmera, os padrões de cores que o Microsoft®Surface

1.0 transmitirá, através de sua superfície, e fazer handshake (conexão) entre ambos (MICROSOFT® SURFACE®, 2009).

A partir disso, todo o conteúdo do dispositivo móvel estará disponível no Microsoft®Surface 1.0 e a interação entre ambos se dará de forma efetiva, como cópia de fotos, músicas, lista de contatos entre outros (MICROSOFT® SURFACE®, 2009).

O fluxograma demonstrado na figura 19, mostra como o Microsoft®Surface 1.0 consegue identificar objetos do tipo *tagged*, colocando-os diretamente sobre sua tela.

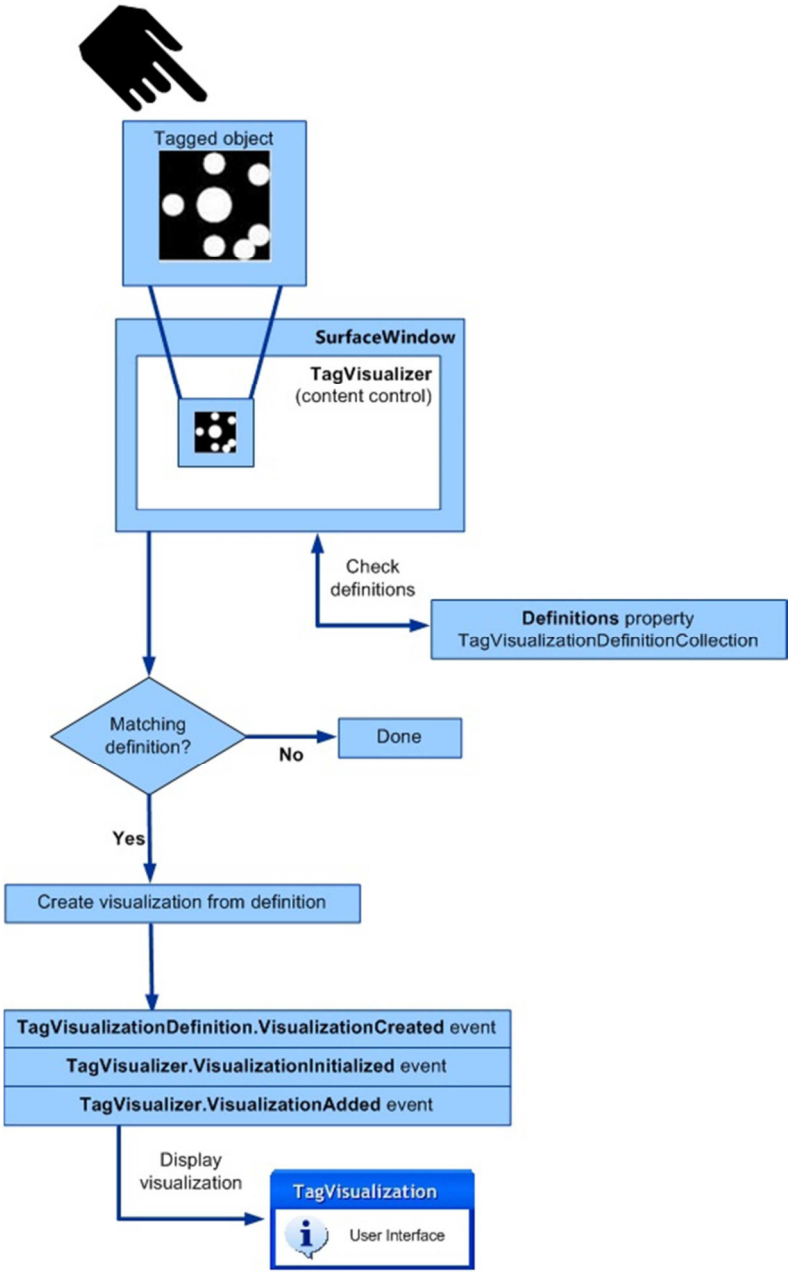


Figura 19 – Fluxograma Microsoft Surface 1.0 SDK

Fonte: MICROSOFT® SURFACE® SDK, 2009

Quando o Microsoft®Surface 1.0 determina que um objeto marcado corresponde a uma das definições estabelecidas, a visualização é criada a partir dos detalhes da definição. Durante este processo, o aplicativo pode fazer manipulação de diversos eventos diversos (MICROSOFT® SURFACE® SDK 1.0 SP1).

4.3 Microsoft®PixelSense 2.0 SDK

O Microsoft®Surface 2.0 SDK (nome referente ao SDK do Microsoft®PixelSense) fornece as API de administração e as ferramentas necessárias para desenvolver as aplicações de superfície. O desenvolvimento de aplicações é essencialmente o mesmo que o desenvolvimento de aplicações WPF ou XNA – framework que serve para o desenvolvimento de jogos para computadores pessoais (PC) com Windows®, para o console Xbox™ 360 e para Windows® Phone 7 – exceto que o SDK fornece suporte estendido para as características especiais do ambiente de superfície como 50 pontos de toque simultâneos, reconhecimento de *blobs*, objetos do tipo *tagged*, detecção da orientação de toques, display na vertical, rotação de tela para exibição, controles especializados entre outros. (MICROSOFT® SURFACE® 2.0).

Para a instalação correta e utilização do SDK, são necessários configurações mínimas de hardware e software obrigatórios, conforme demonstram as tabelas 10, 11 e 12.

Tabela 10 – Requerimentos de hardware para o Microsoft®Surface 2.0 SDK
Fonte: MICROSOFT® SURFACE® 2.0, 2012

Monitor	Memória	Disco	Placa Gráfica
Resolução de 1280x960 ou Widescreen com resolução de 1440x900	4GB ou mais de memória RAM	10GB ou mais de espaço livre no HD	Placa compatível com Aero e 256MB ou mais de memória de vídeo

Tabela 11 – Requerimentos de hardware para o Microsoft®Surface 2.0 SDK
Fonte: MICROSOFT® SURFACE® 2.0, 2012

Sistema Operacional	Service Pack	Ferramentas
Windows® 7 Home Premium, Professional e Ultimate	Windows® 7 SP1	Microsoft Visual C#® 2010 Express Edition ou Microsoft® Visual Studio® 2010, .NET Framework 4. e Microsoft XNA® Framework Redistributable 4.0

Tabela 12 – Programas requeridos para o Microsoft®Surface 2.0 SDK
Fonte: MICROSOFT® SURFACE® 2.0, 2012

Programa	Tamanhos de Instalação	Compactação
Microsoft® Visual Studio	607MB	.ISO
Microsoft® Visual C# 2010 Express	90MB	WEBINSTALL
Microsoft® XNA Framework Redistributable 4.0	48.8MB	.EXE
Microsoft® Surface SDK 2.0	44.2MB	.EXE
Microsoft® Surface Runtime	2.66MB	.EXE

Ao contrário do Microsoft®Surface 1.0 SDK, o Microsoft®Surface 2.0 SDK não gera uma janela de simulação. O próprio monitor do computador onde o SDK está instalado torna-se a tela virtual do equipamento. O SDK instala três ferramentas para o desenvolvimento dos aplicativos:

a) Input Simulator;
b) Input Visualizer;
c) Surface Stress.

A figura 20 ilustra as três ferramentas rodando na área de trabalho do PC onde o SDK foi instalado para estudo e desenvolvimento.

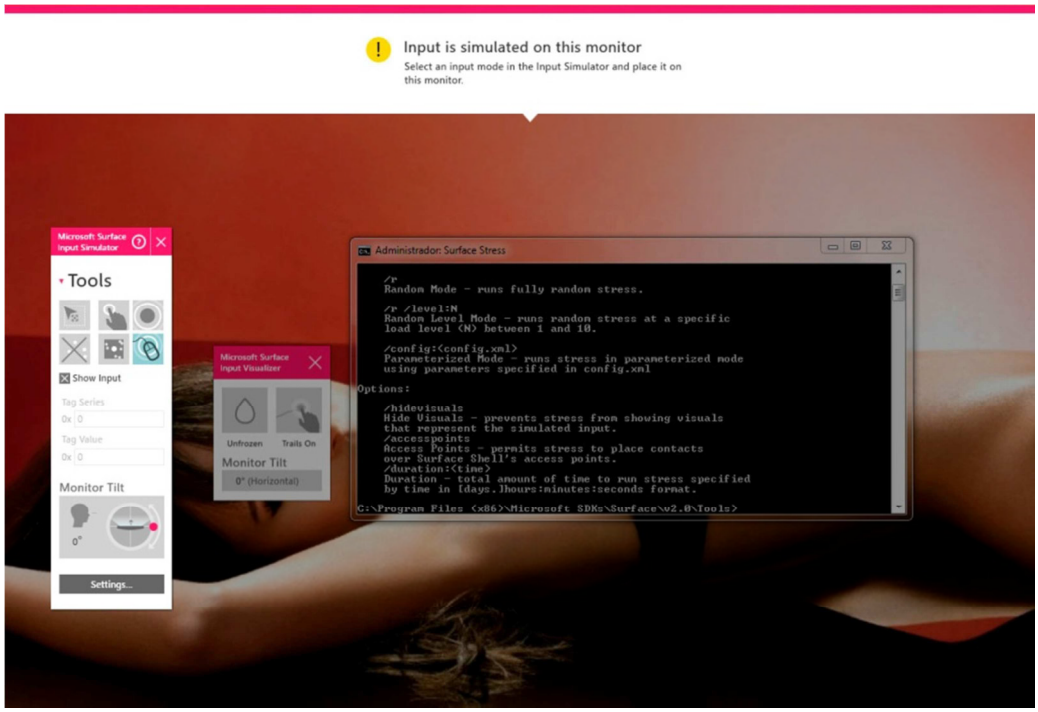


Figura 20 – Interface Microsoft®Surface 2.0 SDK

4.3.1 Input Simulator

Utiliza-se o Input Simulator quando o monitor do computador não tiver a configuração de uma TM. O simulador é aberto a partir do menu Iniciar e executado na área de trabalho. A resolução de tela deve ser definida para 96 pontos por polegada (DPI) antes de iniciá-lo.

O Input Simulator usa um dispositivo de entrada virtual para fornecer dados para o sistema. Enquanto o dispositivo digitalizador virtual estiver ativado, o sistema configurará as opções de capacidade do hardware e receberá os dados de entrada do dispositivo digitalizador virtual.

A figura 21 demonstra como é a ferramenta fica ativa na área de trabalho e seu posicionamento em relação à tela.

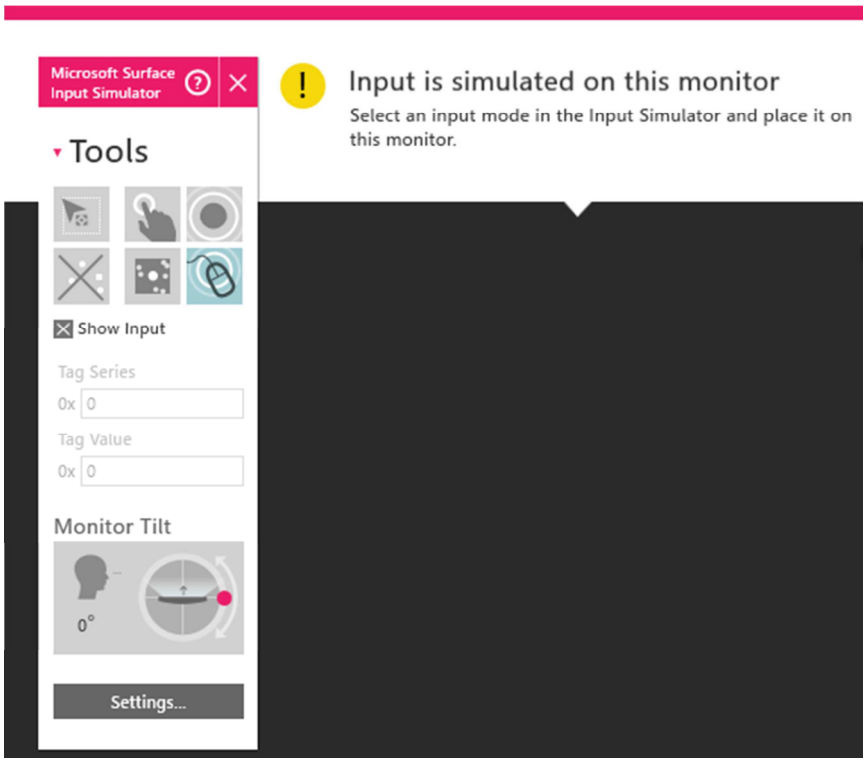


Figura 21 – Input Simulator Microsoft®Surface 2.0 SDK

4.3.2 Input Visualizer

O Input Visualizer é a ferramenta para ver os dados de entrada de toque que o dispositivo de superfície interativa retorna no contexto de sua aplicação. Essa ferramenta é executada em cima da aplicação desenvolvida in loco e exibe informações sobre os pontos de contato que o sistema de entrada detecta.

Esta ferramenta roda somente em dispositivos feitos para o Microsoft®PixelSense ou computador cujo monitor seja do tipo TM. Uma observação importante é que o Input Visualizer e o Input Simulator não funcionam em conjunto.

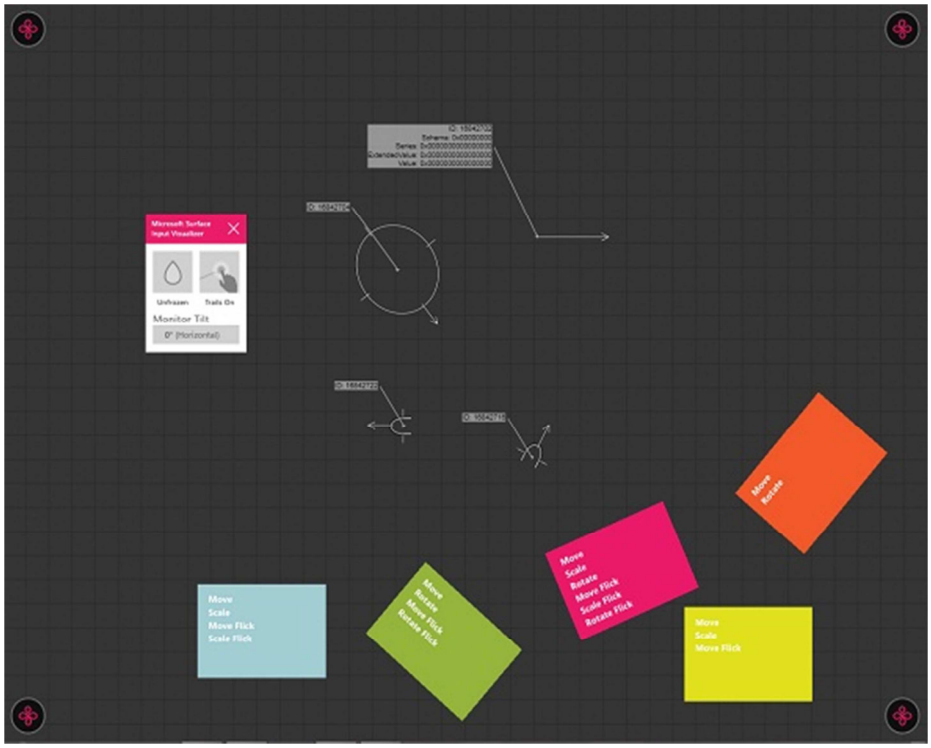


Figura 22 – Input Visualizer Microsoft Surface 2.0 SDK

Fonte: MICROSOFT® SURFACE® 2.0, 2012

4.3.3 Surface Stress

A ferramenta Surface Stress permite testar a estabilidade e robustez do aplicativo, oferecendo simultâneas toques para a aplicação de forma aleatória. A ferramenta gera três tipos de entrada de toque: dedos, *blobs* e *tags*.

Surface Stress é uma aplicação de linha de comando e pode ser executada em dois modos:

a) **Random Mode**: oferece todos os tipos de entrada de toque para a aplicação, com características que variam aleatoriamente.

b) **Parameterized Mode**: permite especificar características de modo de estresse para um controle mais preciso do estresse realizado. Há como especificar um arquivo de configuração e existem arquivos de exemplos de configuração na pasta C:\ Program Files\Microsoft SDKs\Surface\v2.0\Tools\ SampleConfigs.

Assim como o Input Visualizer, Surface Stress não pode ser utilizado em conjunto com Input Simulator.

A tabela 13 informa como executar os dois modos da ferramenta, assim como seus parâmetros.

Tabela 13 – Execução e Parâmetros Surface Stress
Fonte: MICROSOFT® SURFACE® 2.0, 2012

Modo	Linha de Comando	Parâmetros
Random Mode	SurfaceStress.exe /r [/level:N] [/accesspoints] [/duration:Time]	/level: N (de 0 a 10) /accesspoints /duration: Time (hora:minuto:segundo)
Parameterized Mode	SurfaceStress.exe /config:MyConfig.xml [/accesspoints] [/duration:Time]	/config:MyConfig.xml (obrigatório) /accesspoints /duration: Time (hora:minuto:segundo)

5 PROPOSTA DE APLICATIVO MULTITOQUE

Na proposta de aplicativo multitoque considerada neste trabalho, foi feita a modelagem de um aplicativo para TM, de autoria de Sams (2011), a fim de demonstrar o poder IHC através de uma superfície de toque e aproveitar a melhor característica das TM, que é a capacidade de toques simultâneos.

Tornou-se necessária uma abordagem de concepção de qual tipo de aplicativo utilizar. Após análise prévia, o conceito adotado foi o Collaborative Information Retrieval (CIR).

O aplicativo tem como objetivo principal oferecer a manipulação dos dados diretamente na tela do equipamento, onde vários usuários possam interagir simultaneamente e disponibilizar uma visualização correta desses dados.

Através do estudo bibliográfico das características de hardware das TM, optou-se por utilizar o equipamento da PQ Labs™ e seu SDK como parâmetro.

5.1 CIR e Multitoque

O conceito CIR envolve busca, recuperação e visualização de dados como um grupo de pessoas, utilizando uma fonte de dados. (HANSEN, 2005).

A tecnologia de multitoque está em constante redução de custos, o que aumenta sua utilização em diversos ambientes. Sua característica multiusuário sugere que a interação de multitoque pode ser uma possível solução para o conceito CIR (NORTH *et. al.*, 2009).

Sabe-se que ao utilizar os SDK fornecidos pelos fabricantes de hardware, gera-se uma desvantagem, pois as aplicações escritas funcionarão apenas na plataforma de hardware específica, não favorecendo uma migração fácil a outro tipo de plataforma. Por isso, desenvolvedores de aplicativos precisam considerar o tipo de arquitetura do equipamento, ou seja, se este pode suportar o tipo de interação adotado pelo CIR (KHANDKAR, 2010).

O conceito CIR utiliza as TM como dispositivo de entrada e saída de forma bastante satisfatória. Uma mesma superfície torna-se uma opção atraente para a interação de um aplicativo CIR. Computadores de acesso público, caixas automáticos e quiosques de informações são exemplos pela interatividade, facilidade e simplicidade de uso das TM (ALBINSSON *et. al.*, 2003).

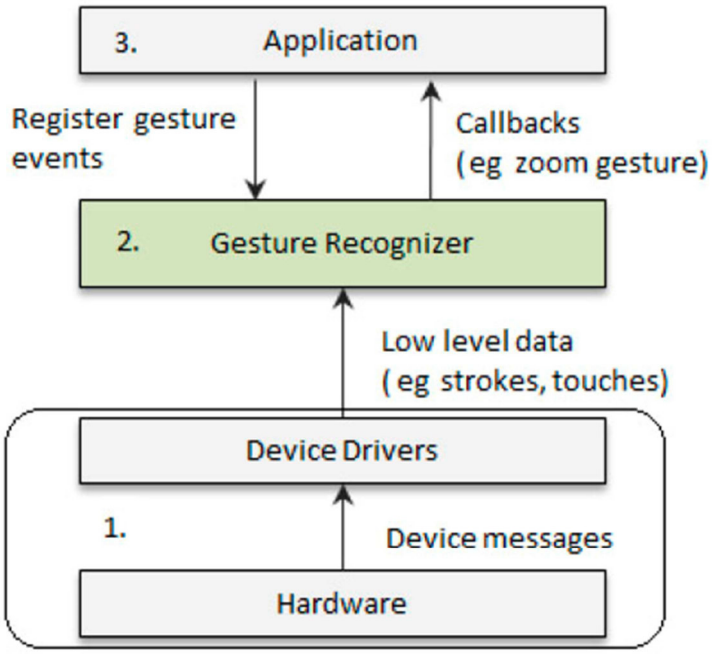


Figura 23 – Exemplo de reconhecimento de gestos dos sistemas multitoque
Fonte: KHANDKAR, 2010

Conforme a figura 23 ilustra, viu-se que o aplicativo escrito para TM requer um mecanismo de reconhecimento de gestos, que tem eventos de gestos registrados nele. O motor de reconhecimento de gestos recebe dados de baixo nível

dos drivers de dispositivo. A partir desses dados, o motor gera mensagens de alto nível, compilando eventos que podem ser repassados para a aplicação via callbacks – pedaço de código executável que é passado como um argumento para outro código, que é esperado para chamar de volta (executar) o argumento em algum momento (LIANG, 1999).

5.2 Biblioteca de Gestos e Eventos

Foi comprovado no estudo dos SDK que não há uma biblioteca de gestos e eventos padronizada. As mais utilizadas são a WPF e TUIO. Diante desse dilema, utilizou-se o GestureToolkit, compatível com WPF para o equipamento da PQ Labs™.

O GestureToolkit permite aos desenvolvedores criar aplicações multitoque sem ter que criar o mecanismo de reconhecimento de gestos ou algoritmos de código de reconhecimento. Quatro definições de gesto, abrangendo manipulação básica de objetos, estão incluídos no GestureToolkit: zoom, girar, arrastar e laço. O GestureToolkit também lida simultaneamente com as interações, distribuição de evento e de gestos para os vários elementos da interface multiusuário.

5.3 Requerimentos para um aplicativo CIR

Depois de escolher a biblioteca de gestos, precisou-se entender como funciona um aplicativo CIR. Foster (2006) classificou os requerimentos para um aplicativo CIR de acordo com a tabela 14, onde informa a frequência de utilização de requerimentos para desenvolvimento coerente com a proposta.

Tabela 14 – Requerimentos para um aplicativo CIR
Fonte: FOSTER, 2006. Tradução do Autor

Item	Descrição	Frequência
1.	Fornecer acesso compartilhado a um espaço de informação	5
2.	Pesquisar por questionamento colaborativo e filtrar espaços de informação	4

3.	Apresentar ou visualizar os resultados de pesquisa ou de informação compartilhada	5
4.	Navegação colaborativa e recuperação de resultados de pesquisa	2
5.	Abrir documentos para usar como fontes de informação	3
6.	Manipular (por exemplo, ordem ou categorizar) os resultados da pesquisa ou informação compartilhada	2
7.	Atualizar ou adicionar valor a informação compartilhada	4
8.	Atualizar os usuários sobre as ações de outros usuários	3
9.	Permitir a comunicação entre os membros do grupo	2
10.	Ativar divisão da carga de trabalho entre os membros do grupo	3

Após análise de cinco aplicações CIR (Ariadne, SearchTogether, Coagmento, Cerciamo, and Annotate!) e de um panorama sobre o comportamento e a classificação de sistemas de multitoque, pode-se definir três requerimentos básicos (KAMMER *et. al.*, 2010):

a) **Independência de hardware**: os aplicativos criados devem funcionar em vários dispositivos multitoque;
b) **Extensibilidade**: o framework deve permitir aos desenvolvedores a criação fácil e simples de novos gestos e eventos;
c) **Separação das camadas de aplicação**: reconhecimento e processamento de gestos devem ocorrer separados da aplicação lógica e da interface com o usuário.

As funções do aplicativo CIR precisam suportar uma biblioteca reutilizável de objetos multitoque e estes devem preencher os dez requisitos funcionais identificados na tabela 14.

5.4 Proposta de Desenvolvimento

O SDK para TM usa um conjunto de eventos nativos que são gerados com gestos. Sendo assim, se faz necessário o uso de um processador de reconhecimentos de gestos. O SDK gera eventos WPF. Esses eventos são criados a partir da informação do dispositivo e fornecem uma medida da independência de hardware (SAMS *et. al.*, 2011).

A figura 24 mostra um diagrama esquemático da proposta. Existem quatro camadas principais: o hardware, o processador gesto, a aplicação e a interação CIR. As primeiras três camadas correspondem a arquitetura de processamento de sinal descrito na figura 23. O GestureToolkit satisfaz os três requisitos básicos: independência de hardware, extensibilidade e separação de camadas de aplicação (SAMS *et. al.*, 2011).

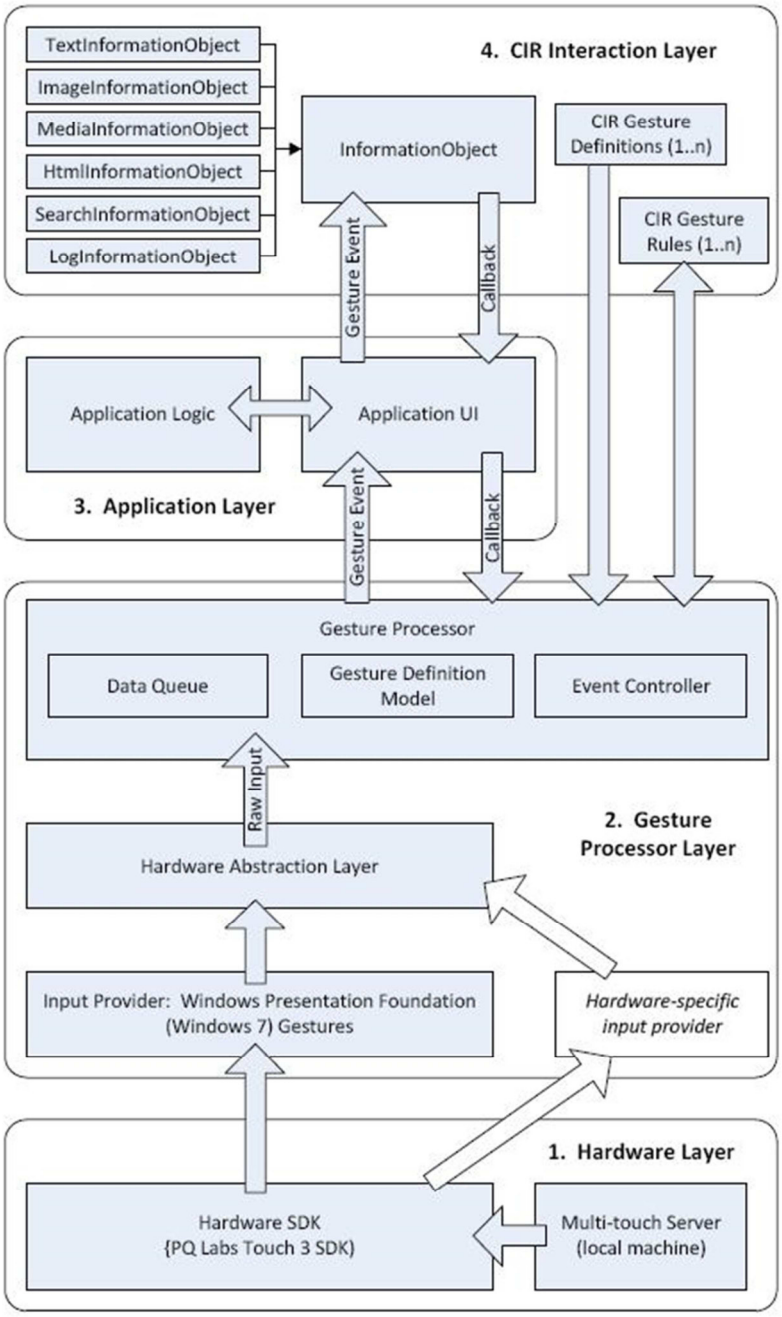


Figura 24 – Diagrama das quatro camadas do esquema proposto

Fonte: SAMS *et. al.*, 2011

Conforme Sams (2011) as definições de gestos e regras são específicas, fazendo parte do framework. Essas definições são carregadas pelo GestureToolkit, garantindo otimização da utilização de memória e poder de processamento. O modelo de definição de gesto é diretamente construído das definições e regras da aplicação CIR.

Sams (2011) ainda explica que o processador de gestos avalia os dados de acordo com a definição de cada gesto. O framework grava cada interação por objeto para permitir a interação simultânea de vários usuários. Depois, compara os dados com um modelo de gesto até achar um compatível e reconhecer o gesto. O processador de gestos calcula os valores de retorno, tais como localização do gesto e o tamanho do mesmo. O controlador de eventos então gera o evento de gesto e direciona a aplicação de interação com o usuário.

Quando a aplicação recebe o evento de gesto, Sams (2011) cita que o objeto de interação é colocado como o primeiro elemento na tela. Este objeto pode ser um documento, uma imagem ou um espaço na área de trabalho. Aplicações CIR precisam prover acesso de informação e visualização dos resultados.

A biblioteca de objetos foi criada como extensão dos controles WPF, obedecendo aos requerimentos do item 3 ao item 7, da tabela 14 de uma aplicação CIR e demonstrada na tabela 15.

Tabela 15 – Objetos de informação e funcionalidades
Fonte: SAMS *et. al.*, 2011. Tradução do Autor

Classe	Descrição
InformationObject	Classe abstrata de funcionalidades: Selecionar, Mover, Zoom, Rotacionar, Duplicar, Anotar, Criar Link, Fechar.
TextInformationObject	Informações de texto: Selecionar Texto, Copiar Texto.
ImageInformationObject	Manipulação de imagem: Cortar.
HtmlInformationObject	Informação no formato HTML: Selecionar Texto, Copiar Texto, Abrir um Link.
MediaInformationObject	Informação áudio/visual: Play/Pause, Avançar, Retroceder, Volume, Tirar Snapshot
SearchInformationObject	Informação de pesquisa. Um objeto especial que faça pesquisa: Pesquisa, Abrir Resultado, Fechar Resultado.

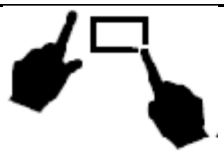
LogInformationObject	Informação de registro. Um objeto especial que registre as ações dos usuários: Desfazer, Refazer.

A classe base é InformationObject. Ela provê atributos e métodos comuns para todos os tipos de objetos. Essa classe é estendida para os quatro tipos de dados e aos dois objetos especiais. Os quatro tipos de objetos são Texto, Imagem, Áudio/Visual e HTML.

A tabela 16 mostra a lista de funções disponíveis, a classe, a descrição da classe e uma imagem do gesto que ativa a função. Convenções de gestos já existentes foram usadas para manipulação básica como Mover, Zoom, Rotacionar e Selecionar. Para objetos específicos, gestos com as duas mãos foram usados, ou seja, enquanto uma mão segura o objeto no local, a outra faz o gesto.

Tabela 16 – Gestos, funções e classes do CIR
Fonte: SAMS *et. al.*, 2011. Tradução do Autor

Função	Classe	Gesto	Imagem
Selecionar Um	Principal	Single tap no objeto	
Selecionar Vários	Principal	Laço ao redor dos objetos	
Mover	Todas	Arrastar o objeto	
Zoom	Todas	2 dedos para movimento de afastar ou juntar	
Rotacionar	Todas	2 dedos para rotacionar	

Duplicar	Todas	Double tap no objeto	
Anotar	Todas	Circular o objeto	
Criar Link	Todas	Arrastar o objeto para cima de outro	
Fechar	Todas	Movimento de arranhar no objeto	
Selecionar Texto	Texto e HTML	Segurar com uma mão e selecionar com a outra	
Copiar	Texto e HTML	Segurar com uma mão e single tap com a outra	
Abrir Link	HTML	Segurar com uma mão e double tap com a outra	
Cortar	Imagem	Segurar com uma mão e desenhar um quadrado com a outra	
Play/Pause	Áudio/Visual	Single tap no objeto	
Avançar, Retroceder	Áudio/Visual	Segurar com uma mão e arrastar para direita ou esquerda com a outra	
Volume	Áudio/Visual	Segurar com uma mão e arrastar para cima ou para baixo com a outra	

Tirar Snapshot	Áudio/Visual	Segurar com uma mão e single tap com a outra	
Pesquisa	Principal	Double tap para criar uma pesquisa	
Abrir Resultado	Pesquisa	Single tap no resultado	
Fechar Resultado	Pesquisa	Movimento de arranhar no resultado	
Desfazer	Registro	Arrastar para a esquerda	
Refazer	Registro	Arrastar para a direita	

5.5 Proposta de Hardware

O equipamento utilizado como parâmetro para a proposta de desenvolvimento foi o PQ Labs™. Foram levados em conta o suporte a biblioteca de gestos WPF, a compatibilidade com o GestureToolKit, possuir um SDK com uma grande variedade de linguagens de programação, em sua maioria de domínio aberto, a compatibilidade com os sistemas operacionais mais comuns no mercado, a capacidade de reconhecer até 32 toques simultâneos e o baixo custo em relação aos equipamentos da Microsoft.

O hardware completo teria em sua composição os seguintes itens:

a) Equipamento PQ Labs™, do tipo overlay de 42 polegadas;
b) Televisão de mesmo tamanho de tela PQ Labs™, para ser utilizada como monitor, montada tanto na vertical como na horizontal;
c) PC para desenvolvimento, instalação e utilização do aplicativo CIR.

6 CONCLUSÃO

Após o presente estudo dos três equipamentos com suas respectivas abordagens de TM disponíveis no mercado, conclui-se que estas facilitam a compreensão da manipulação de dados (textos, mídias entre outros) e impulsionam o uso mais racional, prático e divertido dos novos equipamentos.

As TM possuem diversas formas de implementação e podem ser direcionadas para um tipo específico de aplicação, onde a otimização do equipamento é indispensável, nos quesitos de desempenho, qualidade de tela, desenvolvimento de software e instalação.

Observou-se que o equipamento da PQ Labs™ é superior no custo-benefício, onde o seu preço é mais acessível, suas ferramentas de desenvolvimento, na sua maioria, são gratuitas e a instalação descomplicada.

O estudo do projeto do aplicativo CIR mostrou-se satisfatório por implementar uma biblioteca de gestos utilizada em larga escala, embora não padronizada, e com softwares de desenvolvimento gratuitos tendo vasto suporte das comunidades de programadores. O fluxograma proposto obedece aos requerimentos exigidos ao conceito de um aplicativo CIR e pode ser aplicado a diversas plataformas, bastando apenas adequar algumas particularidades ao SDK do fabricante.

Como trabalho futuro, a compra do equipamento da PQ Labs™ é de suma importância para se por em prática o desenvolvimento de aplicações voltadas para TM, contribuindo para melhor entendimento, ganho de experiência acadêmica e pesquisa. A interatividade do mundo contemporâneo requer aperfeiçoamentos constantes da tecnologia de IHC e nada melhor como as TM para alavancar esse processo de inovação.

REFERÊNCIAS

ALBINSSON, P; ZHAI, S. **High Precision Touch Screen Interaction**. In Proceedings of CHI'03, Ft. Lauderdale, Florida, USA, 5:105-112. Abril, 2003.

ALIKUTTY, A. **Multitouch A Seminar Report**. School Of Engineering Cochin University Of Science & Technology. Novembro 2008.

ASSIS, P. **Como funcionam as telas sensíveis ao toque (touchscreen)**. Disponível em www.tecmundo.com.br/projetor/2449-como-funcionam-as-telas-sensiveis-ao-toque-touch-screen-.htm. Acesso em 06 de novembro de 2012.

BUXTON, B. **Multi-Touch Systems that I Have Known and Loved**. Microsoft Research. Disponível em http://www.billbuxton.com/multitouchOverview.html. Acesso em 23 de janeiro 2013.

CHABAN, A; AKERBERG, H. **Seamlessly access and share data between interactive devices based on natural user interface**. Tese, University os Gothenburg.Janeiro 2012.

COSTA, A. **Superfícies Multitoque**. Instituto Superior Miguel Torga. 2010.

EIZO®. **How can a screen sense touch? A basic understanding of touch panel**. Disponível em: http://www.eizo.com/global/library/basic _understanding_of_touch_panel. Acesso em 17 de dezembro de 2012.

FOSTER, J. **Collaborative Information Seekingand Retrieval**. In Annual review of information Science and technology. 40:329-356. B. Cronin (Ed.), InformationToday, Inc. 2006.

GESTUREWORKS. **True Multitouch for Flash and Flex**. Disponível em http://gestureworks.com/features/open-source-gestures/. Acesso em 17 de dezembro 2012.

GONÇALVES, V; TOMÁS, T. **Interactive Surfaces**; Revista da Universidade de Madeira. P 1-4, 2012

HAJAD, M; ASMA, S. **Seminar of Multitouch Screen Technology**. King Khalid University. 2008.

HAN, J. Y. **Frustrated Total Internal Reflection - Algorithm for multi-touch interface**. Disponível em http://www.cs.nyu.edu/~jhan/ftirsense/. Acesso em 10 de novembro de 2012.

HANSEN, P; JÄRVELIN, K. **Collaborative information retrieval in na information-intensive domain**. Information Processing and Management, 41 (5):1101-1119. Pergamon Press, Inc. 2005.

HARTSON, H R; HIX, D. **Developing user interfaces, ensuring usability through product & process**. John Wiley & Sons, inc. 1993.

JIALIANG, Y; FERNANDO, T; HONGXIA, W. **A multi-touch natural user interface framework**, Systems and Informatics (ICSAI), 2012 International Conference on , vol., no., pp.499-504, 19-20. 2012.

KALTENBRUNNER, M; BOVERMANN, T; BENCINA, R; COSTANZA, E. TUIO - **A Protocol for Table-Top Tangible User Interfaces**, Proceedings of the 6th International Workshop on Gesture in Human-Computer Interaction and Simulation. 2005.

KAMMER, D; KECK, M; FREITAG, G; WACKER, M. **Taxonomy and overview of multi-touch frameworks: Architecture, scope and features**. In Proceedings of the EICS ’10 workshop on Engineering Patterns for Multi-Touch Interfaces, 2010.

KHANDKAR, S; MAURER, F. **A Domain Specific Language to Define Gestures for Multi-Touch Applications**. In Proceedings of the 10th SPLASH Workshop on Domain-Specific Modeling, Reno/Tahoe, 2010.

KUMPARAK, G. **Look out, Microsoft Suface - The iTable might just trump you in every way**. Disponível em: http://www.crunchgear.com/2009/01/10/look-out-microsoft-surface-the-itable-might-just-trump-you-every-way. Acesso em 20 de outubro de 2012.

LIANG, S. **The Java Native Interface: Programmer's Guide and Specification**. Addison-Wesley Professional, 1999.

MICROSOFT® PIXELSENSE®, Disponível em http://www.microsoft.com/ em-us/pixelsense/default.aspx. Acesso em 24 de janeiro de 2013.

MICROSOFT SURFACE. Disponível em http://www.microsoft.com/surface/en/us/default.aspx. Acesso em 14 de dezembro de 2012.

MICROSOFT® SURFACE® 2.0**. Administrador Guide**. Microsoft Press. Disponível em http://www.microsoft.com/. Acesso 23 de fevereiro de 2012.

MICROSOFT® SURFACE® 2.0. **Design and Interaction Guide. Principles & Guidelines for Designing and Developing Surface Applications**. Microsoft Press. Disponível em http://www.microsoft.com/. Acesso 23 de fevereiro de 2012.

MICROSOFT® SURFACE® 2.0. **Developing Applications For Microsoft Surface 2.0**. Microsoft Press. Disponível em http://www.microsoft.com/. Acesso 23 de fevereiro de 2012.

MICROSOFT® SURFACE® 2.0.**Tagged Object Integration For Surface 2.0**. Microsoft Press. Disponível em http://www.microsoft.com/. Acesso 23 de fevereiro de 2012.

MICROSOFT® SURFACE® SDK 1.0 SP1. **Programmer's Guide**. MSDN Library. Disponível em: http://msdn.microsoft.com/en-us/library/ee804954(v=surface.10).aspx. Acesso em 30 de janeiro 2012.

MICROSOFT® SURFACE® SDK 1.0 SP1. **Release Notes.Workstation Edition** Version: October 30, 2009. Disponível em: http://www.microsoft.com/en-us/download/details.aspx?id=15532. Acesso em 30 de janeiro 2012.

MICROSOFT® SURFACE® SDK 2.0. **What's New in Surface 2.0**. MSDN Library. Disponível em http://msdn.microsoft.com/en-us/library/ff727815.aspx. Acesso 23 de fevereiro de 2012

NCR CORPORATION. **Touch Technology Overview**. Disponível em: http://www.computerizedbusinesssolutions.ca/sites/cbs/siteadmin/documents/NCR-TouchTech3_WP_v_1.pdf. 2007.

NORTH, C; DWYER, T; LEE, B; FISHER, D; ISENBERG, P; ROBERTSON, G; INKPEN, K: **Understanding Multi-touch Manipulation for Surface Computing**. In Proceedings of Interact 2009, Uppsala, Sweden.

PETER, B; FLORIAN, D; FLORIAN E; OTMAR, H; JONATHAN, H; MARKUS, L; NIMA, M; LAURENCE, M; PATRICK, O; TIM, R; ULRICH, V. **Multi-Touch Surfaces: A Technical Guide**. Technical Report. P1-19, 2008.

PINTO, P. **Framework para Estação Multitoque de Grandes Dimensões**. Dissertação. Faculdade de Engenharia da Universidade do Porto. Julho 2011.

PQ LABS©, Disponível em http://multi-touch-screen.com. Acesso em 23 de janeiro de 2013

PQ LABS©Multi-Touch **SDK Reference** V1.3.7. Disponível em http://multi-touch-screen.com/sdk.html. Acesso em 20 de outubro de 2012

PQ LABS©Multi-TouchScreen **G2 SDK Reference** V1.2. Disponível em http://multi-touch-screen.com/sdk.html. Acesso em 20 de outubro de 2012

RIBEIRO, J. **Using the PQ Labs Multi-Touch G² 42 inch overlay**. Disponível em: http://justinribeiro.com/chronicle/2009/08/06/using-the-pqlabs-multi-touch-g2-42-inch-overlay. Acesso em 22 de janeiro de 2013.

SAMS, I; WESSON, J; VOGTS D. **An Architecture to Support Multi-Touch Collaborative Information Retrieval**. Nelson Mandela Metropolitan University. 2011

SHNEIDERMAN B. **Designing the User Interface: Strategies for Effective Human-Computer**. Interaction Addison Wesley March 2004 672p. Ed: 4, 2005.

SILVA, M. **Touchboard: Trabalho colaborativo sobre superfícies Multi-toque. Dissertação**. Universidade Técnica de Lisboa. Junho 2011.

SCIENTIFIC AMERICAN MAGAZINE, edição julho de 2008.

SOETENS, G. **Estimating the limitations of the single-handed multi-tuch input**. Tese, Utrech University, Setembro, 2012.

SOUZA, M. **Guidelines usabilidade: discussões para uma abordagem em interfaces multi-touch**. Monografia. Lavras, 2010.

STUMPE, B. **Experiments to find a manufacturing process for an x-y touch screen**. The European Organization for Nuclear Research. p 1-4, Janeiro 1978.

SCHONING; J. et. al. **Multi-touch surfaces: A technical guide**. Disponível em: http://ar.in.tum.de/pub/schoening2008multitouch/schoening2008multitouch.pdf. Acesso em 03 de janeiro de 2013.

TUIO. Disponível em http://www.tuio.org. Acesso em 05 de fevereiro 2013.

WALKER, G. **Display Week 2012 Review: Touch Technology**. Information Display. p 7 vol.28 Set. 2012

WESTING, B. et al. **Integrating multi-touch in high-resolution display environments**. In State of the Practice Reports. ACM, 2011. p 8.