Scalable Vector Graphics (SVG) based multi-level graphics representation for engineering rich-content exchange in mobile collaboration computing environments

Xiaoyong Su, B.S. Prabhu, Chi-Cheng Chu, Rajit Gadh
UCLA, Department of Mechanical and Aerospace Engineering
420 Westwood Plaza, UCLA, Los Angeles, CA 90095

Abstract

The objective of the current research is to create a novel multi-media computational framework that facilitates mobile collaborative engineering field service. In this paper, we propose a 2D graphics hierarchical representation framework and an on-demand content delivery mechanism. Multi-level graphics content sub-division is utilized to transform large engineering graphics, such as drawings, schematic, plans, circuits, and maps etc., into multiple levels of Scalable Vector Graphics (SVG) content. The hierarchical structure of the SVG content maintains the relationship between the sub-divided content and is formed during the process of sub-division. The divided content is selectively delivered and rendered on the mobile devices in an on-demand fashion. A prototypical system of the proposed approach is implemented and the performance of the framework is evaluated through a series of tests. The test results show significant improvement of performance when the framework is used for representing various engineering graphics contents in a real-world mobile collaborative engineering environment.

1. Introduction

The adoption of mobile computing and wireless communication technologies is rapidly changing the traditional business operations and practices in enterprises. A variety of mobile working-models are currently being researched in both industry and academia, as mobile ubiquitous computing is perceived to provide the industry with an avenue to do business in a more flexible (mobile) way. For example, Intel announced Personal Client Architecture [1] as early as 2001 to support high-performance applications on ultra low power mobile devices with integrated voice and Internet capability. Similarly, Microsoft has been developing focused mobile computing platforms for various mobile devices [2]. Other organizations such as World Wide Web Consortium (W3C) [3] and Open Mobile Alliance (OMA) [4] are channeling their research and development efforts in standardizing organization, adaptation, security, representation and delivery of mobile content.

Today, most appealing mobile devices support rich multimedia content and are used as personal digital assistants, organizers, communication tools, and entertainment instruments. As more and more device vendors support powerful processors such as XScale/wireless MMX chips [1] and Open Multimedia Application Platform [5], the new generation of mobile devices are becoming popular computing platforms for both enterprises and consumers to run sophisticated applications and services.

Among obvious advances in mobile computing, there are still a number of issues which need to be addressed. Foremost issue being the conflict arising from the need to support ubiquitous mobility and better performance with the limitation on power and computational capability and constraints on user-input methods and screen size. In addition, wireless communication is fraught with problems of low bandwidth, intermittent connection, and signal interference and thus pose a formidable challenge to mobile collaboration computing environments (MCCE) in general and more so when multimedia content transmission and storage is involved [30].

Activities such as field maintenance, disaster recovery and salvation, exploration survey, etc., typically require engineering information exchange over wirelessly connected mobile devices. One of the major research challenges in this scenario is the modeling, delivery and representation of rich-media engineering content in mobile computing environments which are typically bandwidth constrained and limited in computing capabilities. In the case of engineering field service automation, engineers in the field may have to exchange information with each other and download multimedia engineering content from remote servers or peer to peer devices. Thus, a large amount of data will be transmitted over heterogeneous networks – network services with different capabilities from multiple carriers, and be processed on diverse devices – mobile devices with different capabilities and features. The same situation applies on other mobile applications such as traffic and weather reports, mapping, positioning and navigating, multimedia messaging, animation of interactive graphics, entertainment/gaming, etc.

The above communication and data processing problems usually emerge when large data are transmitted over low bandwidth and intermittent wireless network. For the instance of delivering a 2MB DXF file to mobile device through GPRS network, ideally, the maximum bandwidth of the GPRS connection is 171kbps and with this rate, the total download time T_d will is $2000*8/171 \approx 93.6$s. However, the actual download time usually exceeds the theoretical value because the bandwidth available is typically 30-60 kbps based on the device and the implementation of the service by the carrier. This situation potentially leads to service interruptions and erroneous downloads or multiple times of re-try that results in higher costs. Table 1 shows the download time comparison of transmitting data files of different sizes over different networks to give a perspective of their performance.

Table 1 Download Time Comparison

File size / Networks	20KB	200KB	2MB
GPRS (~36Kbps)	4.4s	44s	440s
802.11b (11Mbps)	0.015s	0.145s	1.45s
100M Fast Ethernet	~0	~0	0.16s

In addition, the parsing and rendering processes of large files on the current mobile devices (low memory, small screen,

limited computing capability, etc.) become unbearable inefficient and difficult to use. In general, graphical content is represented in a single monolithic file which could be a CAD/EDA design, a construction layout, an area map, or a pictures, etc. This file needs to be loaded in its entirety into device memory and rendered on the small screen. This requirement would not cause any problem when the files are delivered and represented on fixed computational devices such as desktop computers or workstations, but would be problematic on mobile devices such as PDA or handheld. Table 2 shows the time comparison of displaying JPEG images (raster graphic content) of different sizes on a mobile device (iPaq5450, XScale 400MHz, 64M RAM) and on a desktop system (Compaq Evo D510, 2.4GHz, 512MB RAM).

Table 2 Time comparison of rendering JPEG file on mobile device and desktop system

File size / Devices	17KB	165KB	622KB
Mobile Device	<1s	~3s	~9s
Desktop computer	~0	~0	<1s

Thus, for successful mobile multimedia delivery and representation in field service automation, especially for operations such as collaboration where content-exchange happens frequently, the computing framework will have to come up with a fundamentally new way for content organization, progressive delivery and also have to provide efficient user interface for collaboration interactions.

In this paper, we report a new framework for engineering field service application where rich-content is predominantly 2D (drawings, schematics, layouts, diagrams, etc.) assisted by other media contents including voice, image, etc. The users are able to view/redline/share/modify/update the engineering content collaboratively on mobile devices. Our attempt is to create a Multi-Level 2D vector graphics modeling and representation concept, and a progressive delivery method using standards based Scalable Vector Graphics (SVG) [6] and eXtensible Markup Language (XML) [7] to try to mitigate the twin principal problems of mobile computing viz. low bandwidth and computing power. The proposed framework can be extended to activities such as gaming, advertising, and medical analysis where 2D graphical information is used gainfully.

The rest of the paper is organized as follows. Section 2 describes related previous work. Section 3 explains the system overview of proposed layered 2D vector graphics and delivery concept. Details of layered 2D vector graphics representation scheme forms section 4. Section 5 describes the details of content delivery and interactive operation. The prototype system and its working are explained in section 6. Finally, we summarize the research in Section 7.

2. Related Work

2.1 2D vector graphics formats

Standard 2D vector data exchange formats have been widely used in multimedia 2D graphics and engineering data representation. The existing vector data exchange formats can be divided into two categories based on their content: vector format for engineering content and vector format for shared multimedia web content. The widely used vector formats for engineering include DXF, CGM, DRW, DWG, PCL, HPGL, PCT, and SHP [24, 25]. Popular vector formats for multimedia web content include WebCGM, CMX, WMF, SVG, and VWPG [24, 25].

Drawing Interchange Format (DXF) [8] was first introduced in AutoCAD (a popular CAD software) and is awidely used CAD format for exchanging data. DXF format is a tagged data representation format. Each data element in the file is preceded by a *group code*. A group code's value indicates what type of data element follows. This value also indicates the meaning of a data element for a given object (or record) type. By using group code, the DXF file is organized into sections. The sections in a DXF file include HEADER, CLASSES, TABLES, BLOCKS, ENTITIES, OBJECTS and THUMBNAILIMAGE. The entities in ENTITIES sections contain various 2D and 3D graphics primitives of the drawings or models. The DXF file can represent CAD drawings using those entities and can connect a group of entities together to form a block (such as windows, doors, etc. in an architectural drawing).

Scalable Vector Graphics (SVG) [9, 27], which has been developed by World Wide Web Consortium (W3C), is a relatively recent standard for two-dimensional vector graphics. Since it is an XML-based graphics description language, it inherits the features of XML such as portability and interoperability. SVG has three types of graphics objects: vector graphics, raster graphics (images, symbols and makers) and text. Graphics objects can be grouped, styled and transformed into reusable components [9]. SVG supports rich graphics primitives such as line, polyline, polygon, path, ellipse, circle, rectangle and special effect such as filter, gradient filling, alpha masks, etc. Another important feature that SVG supports is linking which allows objects to be referenced from different files.

SVG can be used for design, GIS and mapping, embedded systems, location-based services (such as traffic and weather reports, mapping and positioning, navigating… etc.), animated picture messaging, multimedia messaging, animation and interactive graphics, entertainment, eCommerce, and user interfaces information. Currently, there are three versions of SVG specifications: SVG 1.2, SVG tiny for mobile phones and SVG basic for PDAs. SVG tiny and SVG basic together are called Mobile SVG.

2.2 Graphics visualization

There exists significant research focusing on visualization of large data sets in large files, but they are primarily focused on non-mobile environments. In general, hierarchical representation [10, 17], server based computing [11, 18] and tree data structure [13, 14, 15] are most commonly used methods.

Abello J., *et al,* [10] present a hierarchical representation for massive multi-digraphs. They propose a method to transform an arbitrary graph into hierarchies by building a hierarchy of multi-digraph layers on top of the input multi-graph. Each layer consists of sets of vertices of previous level. By using the hierarchy structure, they can control the display space efficiently. Jason Leigh and Stuart Bailey [11] propose a methodology for supporting collaborative exploratory analysis of massive data set in tele-immersive environments. In this methodology, a remote rendering server renders the viewpoints as a sequence of stereoscopic images or animations. Then, the

images and animations will be compressed and streamed to the clients. Paula Frederick *et al.,* [12] present their study on improving the performance of visualization of large figures. They use a persistent structure that is composed of an R-tree [13, 15] and V-Tree [14] for storing and retrieving 2D spatial data. Bin Pham and On Wong [28] have analyzed the capabilities and limitations of current handheld devices and discuss the important issues to be considered such as data organization, management, communication, input methods and user interfaces.

Among these methods, hierarchical representation has a major advantage over the other two methods. But most research on this issue does not discuss content transmission and data rendering on mobile device. A pure-server based solution has to maintain a state for each session and needs lots of computational resource on the server side. Such a solution is not suitable for a collaborative environment with large number of nodes. A hierarchical tree representation could solve the storage and retrieval problem in larger data sets, but it is still difficult for mobile devices to display large data due to limited computing and display power.

2.3 Using SVG standard for engineering data visualization

As a new vector graphics standard, SVG has been widely used for 2D graphics data representation in various fields. Some researchers have used it as visualization language for different types of scientific data. For example, Andres Baravalle, *et al*, [19] use SVG and XSL to visualize dynamically changing data. Yi-Hong Chang, *et al*, [20] use SVG to visualize census data online. C.T. Lewis, *et al*, [21] use SVG to visualize medical data. Other applications include the visualization of GIS [22], human navigation [23] etc.

Sangmi Lee, *et al*., [29] report an SVG based collaborative system, Garnet, for distance-education running on desktops and PDAs. The architecture of Garnet is based on event brokering system. They claim that SVG provides better graphics and document interactivity. They highlight the fact of the possibility of separation of presentation and content layers using SVG, providing flexible viewpoints to the users.

3. System Overview

As mentioned earlier, the prevalent content format in engineering field service automation is 2D graphics. It is usually represented in a single file containing large amounts of information. To use this content, irrespective of its format, it is required to be transmitted and read as a single file in the device memory, and is subject to the bandwidth and computational capability of the MCCE. In this paper, we present a multi-level tree structure of 2D graphics model which is enhanced by multimedia content such voice and images to solve the above problems (which are prevalent in collaborative environments of mobile settings). SVG, the emerging internet graphics standard, is used as the primary file format for the proposed representation framework which allows graphics content to be represented at granular levels of tree structure that enable on-demand delivery and selective display of the graphics content. The term "graphics domain" is used to refer to the set of divided graphics content in this paper.

3.1 2D graphics Multi-Level subdivision

Based on the properties or syntax of 2D graphics content, the primitives that have same or similar properties can be extracted and separated into files that represent different levels. A three-level graphics representation mechanism is introduced for this purpose. Firstly, common properties and attributes such as name space description, coordinate system, etc., are extracted and they constitute the top level of graphics domain description file or the 'Root'. Root file usually does not contain any engineering data. It contains the description of graphics domain and has the embedded links that point to the Level 1 contents. At Level 1, profiles, graphics views, entities groups and graphics objects are extracted. Each type of content may contain engineering data. It also contains embedded links that point to Level 2 contents. Level 2 consists of the detail graphics. It is the lowest level of the content that contains engineering data. The process of dividing single graphics file is defined as multi-level division in this paper.

Table 3 shows the graphic categorization for each level. The "Root", a single description file, is the entry point of graphics content. This file contains information including graphics description (Vendor, Reference, Name Space, etc) and style definition that will be used for all the primitives in this name space except those have been explicitly specified. Level 1 contents include profiles, graphics objects (such as image, symbol, marker and graphics components etc), and views. Level 2 contents include graphics primitives, Annotations, Dimensions, Groups, Blocks, and Text. Criteria for subdivision can be based on primitive type, region of interest, pre-defined templates, or even user definitions.

Table 3 Multi-Level Graphic Contents

Root	Graphics Name Space Description Style Definition, Coordinate System
Level1	Profiles, Graphics Objects, Entities Groups, Graphics Views
Level2	Detail graphics representation, Graphics Primitives, Annotations and Dimensions, Text, Groups, Blocks

When the graphics content is broken into different levels, it should be done in such a way that the level of detail achieved is manageable as well as the granularity is suitable to the MCCE. Too much detail will results in too many small sized files which would need extra bandwidth. The graphics sub-division would provide obvious benefits such as reducing the problems attributed to service interruptions, facilitate short duration transmissions, and only deliver content of interest. Properties will be assigned to each content file to support Level of Detail (LOD) and selective rendering. In the current research, existing tools are used to divide graphics and convert it into SVG format, extra tags for supporting content organization and collaboration have been added to each content file. The graphics decomposition algorithms could be found in elsewhere such as [16].

3.2 Tree structure representation

To support mobile engineering collaboration, the sub-divided contents are then organized into a tree structure. Each

content file is a node of this tree structure. It could have child nodes that are next level contents; or the parent node that it belongs to. In this section, we discuss the process of subdividing single 2D graphics content and organizing the divided contents into a tree structure graphics domain.

If GN represents a graphics domain, $LN1_i$ represents an object in Level1, $LN2_{ij}$ represents the sub-components of object $LN1_i$ and symbol $\oplus$ represents content merging, the following relationships are established:

$$GN = \sum LN1_i = LN1_0 \oplus LN1_1 \oplus \cdots\cdots \oplus LN1_n. \text{ where } i \in (0, n)$$

$$LN1_i = \sum LN2_{ij} = LN2_{i0} \oplus LN2_{i1} \oplus \cdots\cdots \oplus LN2_{im.} \text{ where } j \in (0, m)$$

If we represent this relationship in a tree structure, it can be shown as in Figure 1. In this tree structure, each node represents a single file or directory. We use a graphics domain that has multi-level tree structure to organize these files. For example, if the file name of Level1 node $LN1_0$ is 'PartProfile.svg' (profile of a part in one of the views) and Level2 node $LN2_{01}$ is 'hatch.svg' (hatching of the profile of the part), then the full lookup index of this branch of the tree will be *YOUR_ORGANIZATION.GN.LN1$_0$[PartProfile].LN2$_{01}$[hatch]*. These nodes can be dispersed over a number of machines in an enterprise network. Via this representation, graphics content can be precisely located in a distributed environment.

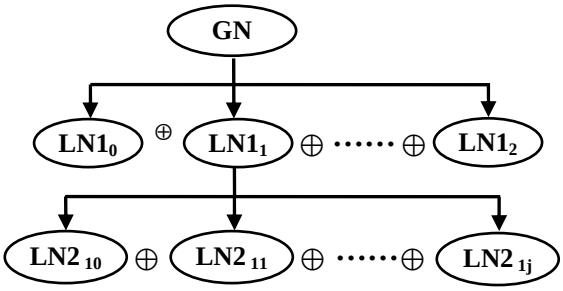


Fig. 1 The tree structure of the multi-level division and content organization

3.3 An instance of Multi-Level division and content organization

Figure 2 shows an engineering drawing containing Multi-Level information. It has three views, textual annotation, drawing information, dimensions, etc., and each of these can be separated into different nodes (files) containing semantically related information.

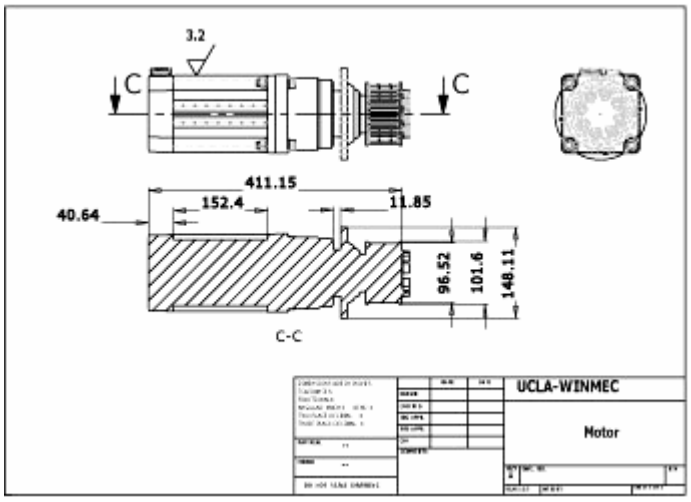


Fig. 2 An engineering drawing with dividable graph information

Firstly, the common properties such as line style, line width, font face and size, etc., are extracted. At the same time, the first level sub-division will be conducted. The graphics content will be divided into sheet, main view, left view and top view.

In the process, common properties, along with the general description of this drawing such as vendor, language etc, will be used to form the root and it will constitute the top level description file. Starting from this file, it is possible to traverse the entire graphics content.

In the second stage, the second level division is carried out. In the case of top view of the above example, the top view has been divided into profile, hatch and dimension.

The original drawing can be regenerated by reversing the whole process. Figure 3 shows the subdivision sequences of this engineering drawing.

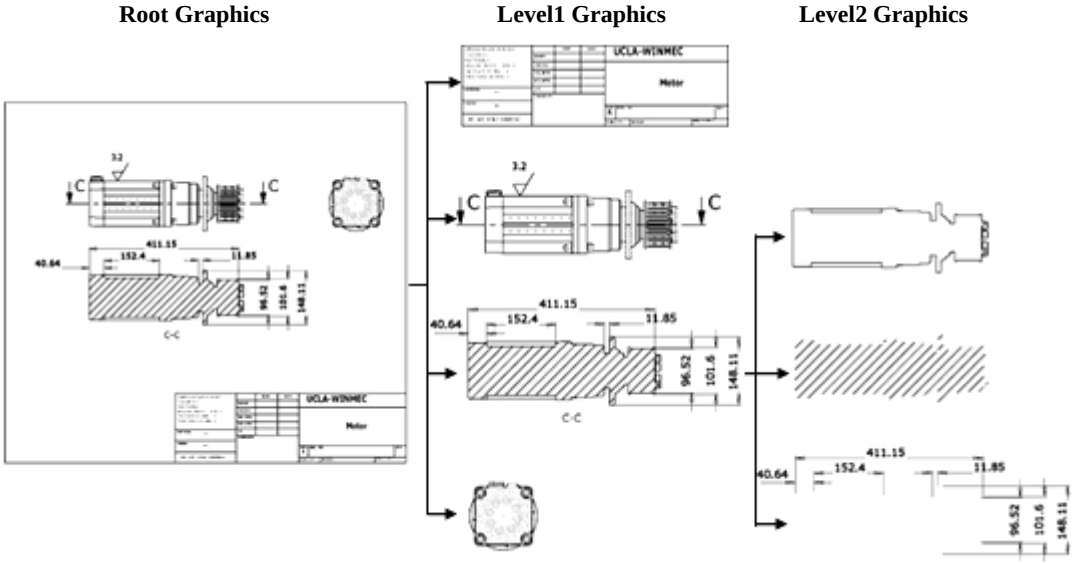


Fig. 3 Sub-division sequence and layout

4. Multi-Level 2D graphics content representation by SVG

4.1 Scalable Vector Graphics (SVG)

Scalable Vector Graphics (SVG), which is based on eXtensible Markup Language (XML), is an open 2D vector/raster mixed graphics standard. It has a well defined file format and a programming API framework. SVG has already gained wide support from industry. A number of free SVG viewers, editors, converters and SVG Development Toolkits are available from a number of companies. The following list comprises of the different features that are supported by SVG:

1. **Graphics objects –** SVG supports three types of graphics objects: vector graphics shapes, images and text. Vector graphics primitives are paths, rectangles, circles, ellipses, lines, polylines and polygons. The properties of graphics primitives can be either set by Cascade Style Sheet (CSS) [31] or individual setting. Coordinate Systems, Transformations are also supported in SVG.
2. **Document structure –** SVG content is organized into Document Object Model (DOM) [32], which is a tree structure containing objects and allows dynamic content access and update. This feature is very important for representing content on a mobile device because the user can selectively choose the content to be rendered.
3. **Special Functionality** – SVG supports linking, interactivity and record events. These functionalities are essential for real-time collaborative activities.
4. **Effect of graphics –** SVG supports special graphics effects including Painting, Gradients, Patterns, Clipping, Masking, Compositing, Filtering, and Animation.
5. **Other features –** Other features that SVG supports include metadata, scripting, multimedia contents, etc.

SVG utilizes a syntactical structure that is similar to XML. Elements of a typical graphic content can be described by a sequence of drawing elements of SVG. The following example gives us an idea of content organization in SVG. It represents a non-filled rectangle, a circle with red color filled and a text.

```
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.0//EN"
  "http://www.w3.org/TR/2001/REC-SVG-
20010904/DTD/svg10.dtd">
  <svg width='200' height='200'>
   <title>A Rectangle, a circle and a text</title>
   <rect x="40" y="40" height="100" width="100" fill="none"
stroke="black" stroke-width="1"/>
   <circle cx='100' cy='100' r='40' style='fill: red;'></circle>
   <text x='60' y='160' fill='blue'>Hi, SVG</text>
</svg>
```

As we can see from this example, besides the graphics description sentences, there are some other descriptions at the beginning of the file. These sentences are necessary for describing document type, styles and other useful information, but they increase the file size. In the current research, minimize these descriptions to contain only the essentials in the graphics domain without violating the requirements of the standard thereby providing maximum portability. In the following sections, we will explain how to use SVG to dissect the above engineering geometry examples.

4.2 Using SVG to describe the Top Level 2D graphics

As described in section 3, 2D graphics content can be divided into several levels. The top level (Root) is the entry point of the 2D graphics. It contains general information of the system such as vendor, time, name space, coordinate system, and style definition etc. This general information is referenced by all the subsequent levels within a given graphics domain. The following is an example of the root level content of a 2D graphics represented by SVG. At this level, general information of the graphics content is defined and Level 1 links are established.

```
<! --------- SVG announcement ---------->
<?xml version="1.0" standalone="yes"?>
<root xmlns:Project1=http://www.wireless.edu/2004/Project1 />
<svg width="600.0" height="391.429" viewBox="0 0 1050.000
685.000" version="1.1" xmlns="http://www.w3.org/2000/svg">
<! ---------- Descriptions ---------->
<title>MotorDrawing</title>
<! – parent is important to establish a two way relationship between
each file -->
<parent>self</parent>
    <desc>
        <Property>general information</Property>
        <Vendor>WINMEC</Vendor>
            <ProductNo>PN123.342<ProductNo>
            <Version>Version 1.0001</Version>
            <Time> JAN-04-2002, 21:05 </Time>
    </desc>
 <! ---------- Styles ---------->
<style type=”text/css”>
……
</style>
<! ---------- graphics ---------->
<g>
            <image xlink:href=Sheet.svg />
            <image xlink:href=TopView.svg />
            <image xlink:href=RightView.svg/>
            <image xlink:href=SectionView.svg/>
  </g>
</svg>
```

There are four parts in this description file. The first part announces a Root SVG file and defines the namespaces. The second part describes the general information of the graphics content layout. Usually, this part is not rendered on the graphics screen, but is used in rendering the lower level objects, however, this information can be displayed when requested. The third part is cascade style sheet definition, which defines all the styles that are used for rendering the entire graphics domain. The fourth part is the graphics structure, which points to files belonging to the next level, in the form of ‘link’ tags of SVG. Thus, the motor drawing is now divided into four separate files.

4.3 Using SVG to describe Level 1 and Level 2 graphics

In Level 1, SVG parameters are announced and graphics structure is defined. Let’s use section view as an example. It has

been divided into three files (hatch.svg, profile.svg, and dimension.svg) based on the primitive's properties. The following code is the SVG representation of the section view.

```
<?xml version="1.0" standalone="yes"?>
<svg width="600.0" height="391.429" viewBox="0 0 1050.000
685.000" version="1.1" xmlns="http://www.w3.org/2000/svg">
<title>SectionView</title>
<parent>MotorDrawing</parent>
<desc>
        <Property>views</Property>
</desc>
<g>
        <image xlink:href=hatch.svg/>
        <image xlink:href=profile.svg/>
        <image xlink:href=dimension.svg/>
 </g>
</svg>
```

In these levels the unique feature of SVG standard of inheriting properties is utilized to maintain contextual relationship between levels. In general, if the content is displayed in standalone format, the current SVG parameters are used. However, if the current content merges into parent level, parent's parameters are used. The graphic structure of Level 1 uses the properties of the Root level in our representation. Here, again, 'links' are used to point to the next level graphics sub-objects or primitives. Section view now has been divided into three files.

Level 2 represents actual graphics details. It consists of graphic primitives which can be grouped and transformed to form the desired graphics content. As mentioned before, the Level 2 contents of the section view contains three files: hatch.svg, profile.svg and dimension.svg. For brevity, only partial content of the hatch.svg file is presented as following.

```
<?xml version="1.0" standalone="no"?>
<svg  width="600.0" height="391.429" viewBox="0.0 0.0 1050.000
685.000">
        <title>hatch</title>
        <parent>SectionView</parent>
   <desc>
      <Property>hatch</Property>
        </desc>
        <g id="group1" fill="none" >
<path d="M270.87 342.7L263.46 350.12M279.83 …… " fill="none"
stroke="black" stroke-width="0.2"/>
……
        </g>
</svg>
```

By using the Multi-Level representation, a large single file has been divided into multiple small files. For instance, the original file size of the example that we are using is 233KB. By doing 'level1' subdivision, it has been divided into rightview.svg, which is 131KB; topview.svg, which is 27KB; sheet.svg, which is 28KB; and sectionview.svg, which is 20KB. And going to 'level2', sectionview.svg has been divided into hatch.svg, which is 2KB; profile.svg, which is 6KB and dimension.svg, which is 13KB.

Effective content organization allows a single user to operate data, or multiple users to simultaneously operate data on a server, or multiple users to share data piecewise when real-time collaboration is required. Another research problem that becomes important in this context is content synchronization which we plan to address in future research.

4.4 Maintain the relationship of the divided contents

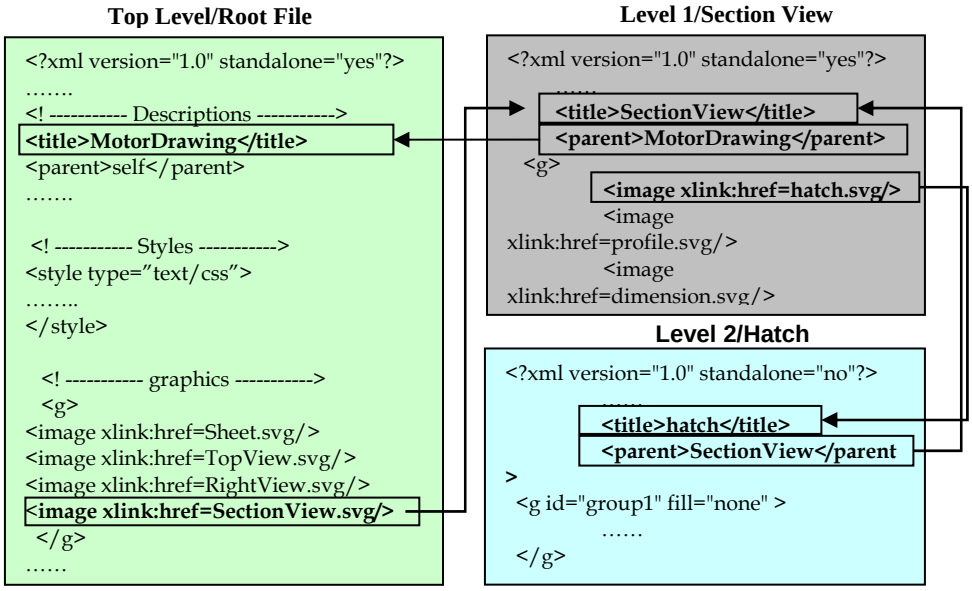


Fig. 4 Pointers in SVG files

In our representation, as shown in the above examples, the two-way relationship between the decomposed contents is maintained by using <parent> element and Xlink attributes. The advantage of this two-way relationship referencing is that the entire hierarchy of the tree can be traversed by starting from any node in the graphics domain. This is particularly efficient on mobile devices because only current active node needs to be loaded into memory. Figure 4 shows the pointers in SVG files. Figure 5 shows the topology of the two-way relationship between each content node.

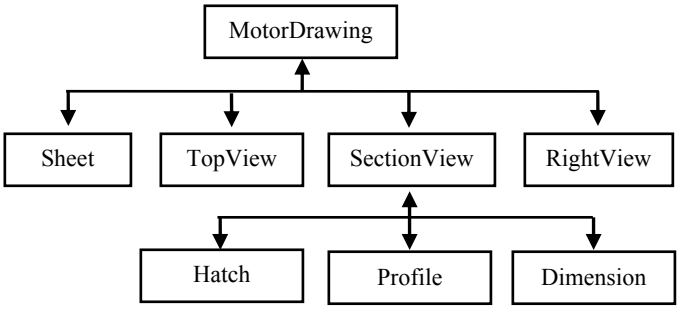


Fig. 5 Two-way relationship topology

4.5 Enhance the SVG content for supporting collaboration

In the mobile collaboration computing environment, engineers may also want to share their opinions with each other on a particular piece of information. Text or voice based comments are typically used in such a scenario. Text or voice comments could be recorded and inserted into the content node and then be uploaded to peer device or server. The advantage of separating the voice into a new file is that the original graphics file would not be modified too much. The following code demonstrates how to add a comment into the section view. More comments can be appended to the head of the file.

```
<?xml version="1.0" standalone="yes"?>
<svg width="600.0" height="391.429" viewBox="0 0 1050.000
685.000" version="1.1" xmlns="http://www.w3.org/2000/svg">
<title>SectionView</title>
<parent>MotorDrawing</parent>
<comment text="the dimension of 411.15 should be 420"
reviewer="Tom"/>
<! – To add voice comments-->
<audio xlink:href="comment1.wav">
<g>
        <image xlink:href=hatch.svg/>
        <image xlink:href=profile.svg/>
        <image xlink:href=dimension.svg/>
</g>
</svg>
```

5. Content delivery and interaction

Sub-divided contents can be delivered in two ways. The first is called 'sequential delivery'. Once the root is loaded in the memory, the system automatically uses the links in the root to discover and download the contents that the links point to. This approach may be utilized in fixed computing environment which has less limitation on computational resource. The second method is 'on demand delivery', where the root is downloaded first while the content of the next level are not. In turn, the system provides a list of all the content of the next level which can be selected as per the user's need. The type of delivery can be chosen by the client as required. The content delivery method in either of the two delivery mechanisms can be controlled during the delivery itself using attributes (such as Level of Detail) that define the content.

The current approach utilizes the 'on-demand' content delivery with interaction architectures. In a more intelligent fashion, the content delivery process can be controlled by device profile or present network status. The World Wide Web Consortium is working with industry on a standard called Composite Capabilities/Preferences Profile [33] to describe device profile so that different platform can exchange device profile or network status in a heterogonous computing environment. A service agent running on the mobile device is used to detect current usable memory, battery level, CPU, screen size, etc. The service agent could also obtain the current network connection status and predict the bandwidth and wireless signal change. During the delivery process, the client's profile will be used to determine which level of content, in what detail, and in which format should the content be delivered. The content delivery implementation in the current prototypical system is based the demand of the operator.

This two-tiered approach of intelligent content download provides a flexible and efficient framework when contents are shared between collaborating users of mobile devices or fixed computing device.

5.1 Graphics content delivery framework

Figure 6 shows the schematic of the current delivery framework. In this framework, content delivery is initiated by client requests. As mentioned earlier, the graphics content has been sub-divided into three levels and a tree structure of the contents is maintained. By subdividing large content into interconnected multiple small files, content server does not need to maintain the state of whole interaction/transaction. Content adaptation and rendering are carried out at the client side.

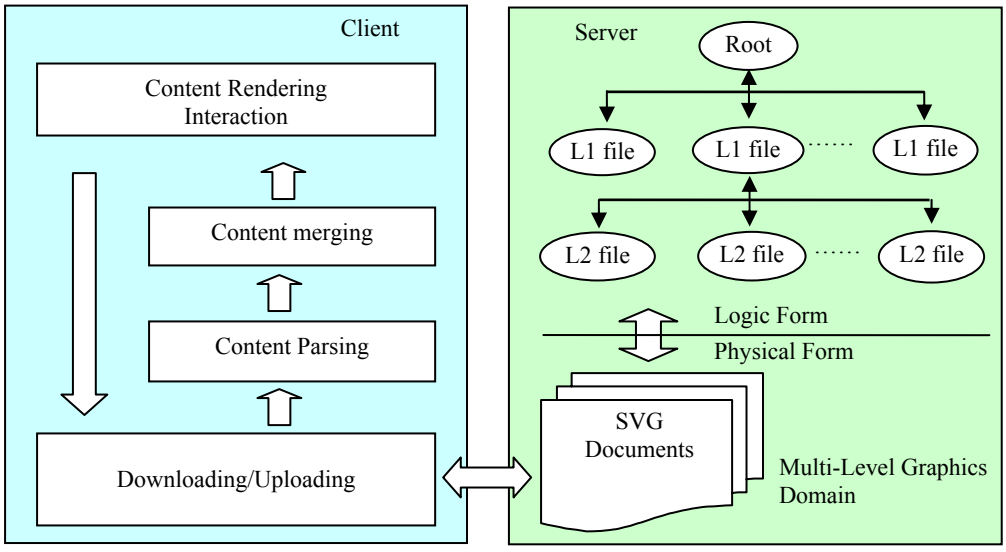


Fig. 6 Graphics content delivery and process (L1 means Level 1 and L2 means Level 2)

Advantage of this delivery method is that the server can support more clients at the same time – a requirement of collaborative design.

The server maintains the graphics domain and listens to clients' requests. A client has four major components: Downloading/Uploading, Content Parsing, Content Merging, and Content Rendering/Interaction. Downloading/Uploading module maintains the conversation, sends content request to server, handles the downloaded content and uploads the changed content. The Content Parsing module will read the content into memory and build a DOM structure for each individual file. Content Merging module interacts with the global DOM structure and merges the downloaded content with the global DOM structure. Two of the Content Merging methods will be discussed later in this paper. Content Rendering and Interaction module enables content visualization and interaction on the client. Through this module, the end user can trigger downloading/uploading, start or end a session, request a portion of content.

5.2 Content merging and rendering

Because multiple files are used to represent a graphics domain, each file has its DOM structure built when the file is downloaded to the client. DOM plays a critical role in content merging. The client maintains a DOM structure for each downloaded file during a session. When the client renders the content to the device screen, the client can selectively choose which node of the content needs to be rendered onto the screen based on the content property. Level of Detail rendering can also be accomplished by applying a set of conditions on the properties. This process can be carried out in three ways.

The first method is via Simple Content Merging. Firstly, client loads one content node. In this content node, links that point to sub-level contents may exist. The user client then loads the linked contents into memory and replaces the links. This procedure does not need to start from the parent because each content node has a link point to its parent. If certain sub-level content is loaded first, the parent content can be loaded by creating a new DOM from the parent content and by inserting the existing DOM into the new DOM. Since each content node contains information on its corresponding properties, the entire content could be selectively rendered based on the client's settings.

The pseudo code to insert the sub-level content to a DOM is as follows:

```
In Parent DOM Structure pDOM
For each elements ei
    If elements type is SVG link
            Then load the SVG DOM Structure sDOM
            pDOM.ei.Replace(sDOM)
    end
end
pDOM.SelectiveDraw()
```

The second method of merging is via Hyper Content Merging. In this method, instead of loading the actual sub-level, a hyper link is created for each sub-content node. End clients control how and when the contents are loaded. This process is similar to that of loading HTML files in web browsers. It is particularly useful when the computational resource is limited because the client does not need to maintain the whole content DOM structure which is generally costly. The pseudo code of this method is as follows:

```
For each elements i
    If elements type is SVG link
            Display the title at proper position (or list the content)
    end
end
```

The third method is called Hybrid Content Merging. This method takes advantages of both methods that mentioned above. It gives the client the flexibilities to load the sub-level contents. The following is the pseudo code of this method:

```
In Parent DOM Structure pDOM
For each elements ei
    If elements type is SVG link
          if bDraw is true
             Then load the SVG DOM Structure sDOM
             pDOM.ei.Replace(sDOM)
          end
    end
end
pDOM.SelectiveDraw()
```

5.3 Interaction

The mode of interaction and collaboration is very important in an engineering field collaborative environment where clients need to add comments, redlines, and other observations to the original graphics content. Text based comments will be inserted into the content node directly as mentioned in section 4.5. Other form of content addition will be stored as a separate SVG file and an entry link in the content file will be created. The Downloading and Uploading Module will send a request to update the graphics domain on the server or pier device so that this change can be reflected on the devices of collaborating partners. This process allows progressive changes to the graphics domain. In the current prototypical system, text comments and voice comments are implemented.

6. Prototypical system and evaluations

6.1 Prototypical system

Based on the framework presented above, a prototypical system is implemented. This system contains two modules. One is content visualization and interaction user client, the other one is mobile collaborative client. Both are implemented on Pocket PCs, iPaq 5450 with Intel PXA 250 400Mhz XScale processor, 64MB RAM and 48MB ROM.

The mobile collaborative client is created as a layer above a message oriented middleware [26]. Mobile user can share documents, access enterprise database, and communicate with group members by using email, chat and voice messages. The basic functions of the system include documents sharing/representation, communication, process monitoring, task assignment, and member management.

Figure 7 shows some screen shots of the current mobile collaborative system. The comments attached to the engineering drawing can be obtained by clicking the comments button in visualization client interface.

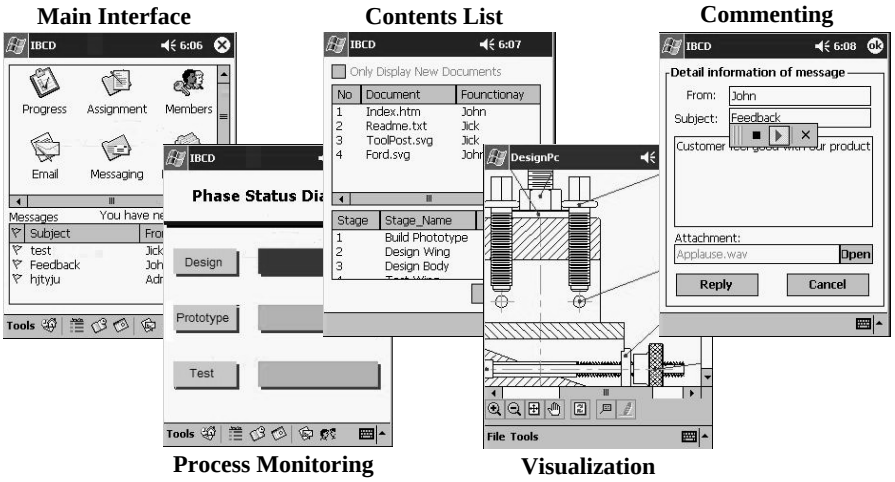


Fig. 7 Screen shots of the prototypical mobile collaboration system

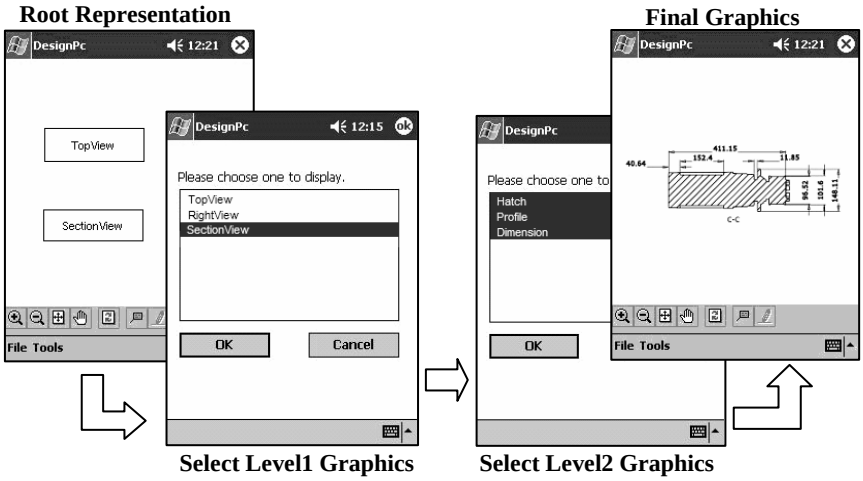


Fig. 8 The selective content rendering process

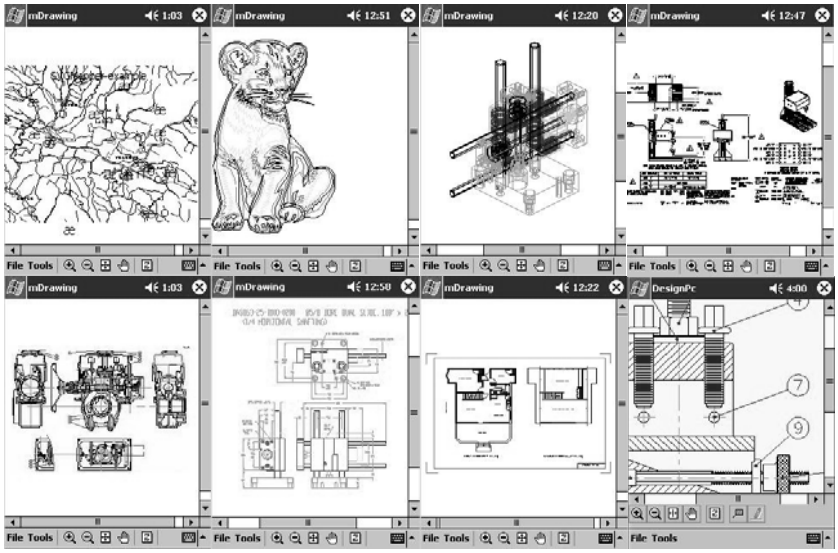


Fig. 9 A few sample graphics used for evaluation

Figure 8 shows the screen shots of the content visualization and client interactions which selectively render pre-organized graphics contents. It allows "zoom in", "zoom out", "zoom all" and "pan move" operations of graphics content. It also provide the ability to add comments to current graphics content, selective display graphics contents, and reload parent content, etc.

Taking the 2D drawing shown in Figure 3 as an example, the drawings of the part form the multiple views in the 2D graphics domain. First, the root content file, which contains no graphics information, is loaded. Second, the client lists all the relevant links of the domain and the user can choose to display which links or views he/she would like to see. In the displayed view, there are chained links which link to another level of drawings. By following the links, users can navigate from the root to the lowest level of the graphics (or called the leaf node of the tree structure) and vice versa.

6.2 Performance evaluation

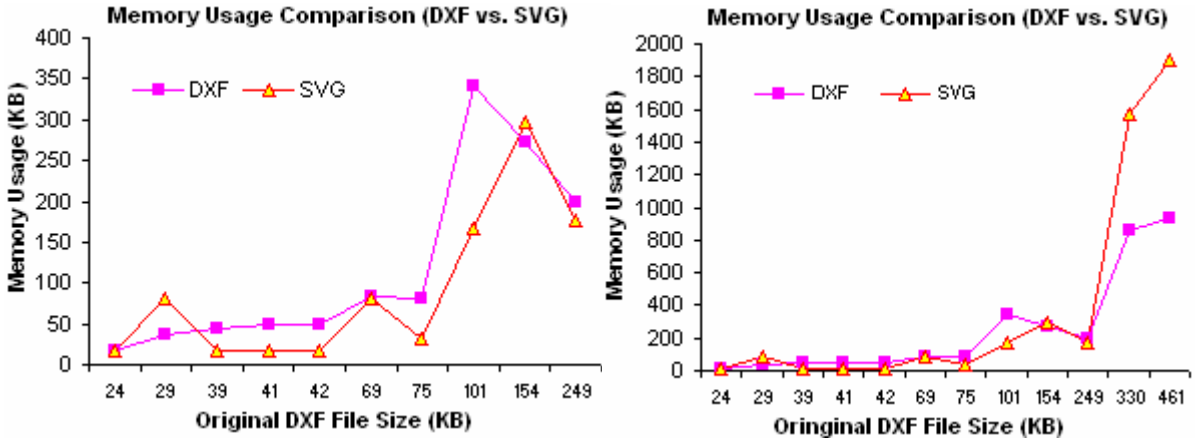
With the prototypical system, the performance of the proposed framework is evaluated in terms of memory usage and user-application experience. The evaluation is based on 802.11b wireless connection. The performance evaluation of wireless wide area network is being planed as the future work of this research. The following results are based on the tests performed on iPaq 5450 which equips with Intel PXA 250 400Mhz XScale processor, 64MB RAM and 48MB ROM.

6.2.1 Memory Usage Evaluation

In Memory Usage Evaluation, 12 different DXF files are converted to SVG files (one SVG file per DXF file) and the memory storage requirement for each file is recorded. Figure 9 shows various sample graphics that are used for performance evaluation. Figure 10 (a) shows the memory utilization comparison when the original DXF file size is less than 250 K. Figure 10 (b) shows the same comparison of all the 12 DXF files.

As indicated in Figure 10 (a), out of the ten DXF files, eight SVG files utilize less memory than DXF files do. This indicates that SVG shows better performance for memory utilization when the file sizes are small (Small file size means simpler drawing). From Figure 10 (b), as the original DXF file size gets

larger, the SVG file occupies more memories than DXF file. This indicates that the SVG format is not as efficient when the file size is large (Large file size means complex drawing). Therefore, to take advantage of SVG, further sub-division of large SVG files is required. This can be accomplished by the proposed approach described in Section 4 of this paper. In addition, in SVG, groups of similar graphics can be represented as a path, thus reducing the size of the converted SVG file.



(a) (b)

Fig. 10 Memory usage comparisons (a) Comparison when file size is less than 250K. (b) Overall comparison.

6.2.2 User Experience Evaluation

The User Experience Evaluation is done by measuring the operation time of loading and zooming of 19 different SVG files on the mobile platform. Figure 11 shows the test results of this evaluation. The response of system is 5 to 10 times faster when a smaller SVG file is rendered. It is also observed that when the system is busy in processing graphics content, the wireless connection of the PDA experiences short period of disconnection.

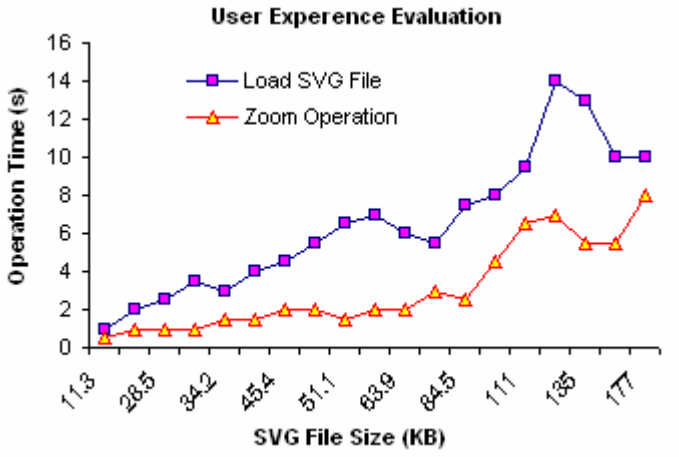


Fig. 11 User experience evaluation

7 Summary

The endeavor of this research is to investigate a new software framework and architecture for multimedia content representation and transmission based on industrial standards so as to mitigate the problems of bandwidth and computing constraints in a mobile and collaborative design environment.

In this research we have proposed an SVG based distributed, hierarchical 2D graphics content representation and progressive delivery architecture to facilitate rich engineering content exchange in mobile collaboration computing environment. The proposed architecture employs a multi-level division method to dissect the graphics content into multiple levels using a variety of rules and strategies. Each level can be delivered on demand in contrast to the traditional approaches, i.e. delivery of the entire monolithic file. A prototypical system of the proposed approach is implemented and used for performance evaluations. The testing shows satisfactory results in representing different engineering graphics content (drawings, schematics, layouts…, etc.). This approach therefore holds a promise for use in mobile and collaborative field service engineering applications.

The prototype system is being developed wherein modules for real-time device based content adaptation, personalized user interactivity, persistent session management and migration, and content synchronization modules are being investigated and implemented. Future research involves studying the scalability of the architecture, adding of other multimedia components, such as video and audio, in the design collaboration, extending the current hierarchical model for video/audio data streaming, continuing the research to include additional types of engineering data such as FEA analysis results, and implementing the collaboration framework via web services.

Acknowledgement

We are pleased to acknowledge the support of Intel Corporation, and Wireless Internet for the Mobile Enterprise Consortium (WINMEC), UCLA towards partial funding for this research work. This research work has been done in the Wireless Media Lab at UCLA and the Mechanical Engineering Department at UCLA.

Reference:

[1] PCA Overview, Intel, http://www.intel.com/pca/developernetwork/overview/index.htm

[2] Microsoft Mobile, Microsoft, http://www.microsoft.com/windowsmobile/default.mspx

[3] World Wide Web Consortium, http://www.w3.org

[4] Open Mobile Alliance, http://www.openmobilealliance.org/

[5] OMAP for 2G and 3G: Overview, http://focus.ti.com/docs/apps/catalog/overview/overview.jhtml?templateId=1106&path=templatedata/cm/level1/data/wire_omap_ovw

[6] Scalable Vector Graphics, W3C, http://www.w3.org/Graphics/SVG/

[7] Extensible Markup Language, W3C, http://www.w3.org/XML/

[8] Drawing Interchange Format, Autodesk, http://www.autodesk.com/techpubs/autocad/dxf/

[9] About SVG, W3C, http://www.w3.org/Graphics/SVG/About.html

[10] James Abello; Jeffrey L. Korn; MGV: A System for Visualizing Massive Multidigraphs. IEEE Transactions on Visualization and Computer Graphics 8(1): 21-38 (2002)

[11] Leigh, J.; Johnson, A.E.; DeFanti, T.A.; Bailey, S.; Grossman, R.; A methodology for supporting collaborative exploratory analysis of massive data sets in tele-immersive environments, Proc. of The Eighth International Symposium on High Performance Distributed Computing, Aug 3-6, 1999

[12] *Frederick, P.; Gattass, M.; Mediano, M.R.*. Efficient visualization of graphical objects, Proc. of XII Brazilian Symposium on Computer Graphics and Image Processing, Oct 17-20. 1999.

[13] Timos Sellis, Nick Roussopoulos, and Christos Faloutsos. The R+-Tree: A Dynamic Index for Multi-Dimensional Objects. In *Proceedings of the 13th VLDB Conference*, pages 507–518, September 1987.

[14] Maur´ıcio Riguette Mediano, Marco Antonio Casanova, and Marcelo Dreux. V-tree - a storage method for long vector data. In *Proceedings of the 20th VLDB Conference*, 1994.

[15] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *Proceedings of the ACM SIGMOD Conference on Data Engineering*, pages 47–56, 1984.

[16] B. S. Prabhu; S. S. Pande. Intelligent interpretation of CADD drawings. Computers & Graphics 23, pages 25-44, 1999

[17] Leila De Floriani, Enrico Puppo. Hierarchical triangulation for multiresolution surface description, ACM Transactions on Graphics (TOG), Volume 14 Issue 4, 1995

[18] Jin Jing, Abdelsalam Sumi Helal, Ahmed Elmagarmid, Client-server computing in mobile environments, ACM Computing Surveys (CSUR), Volume 31 Issue 2, 1999

[19] Andres Baravalle, Marco Gribaudo, Vitaveska Lanfranchi, Using SVG and XSLT for graphics representation, Proceeding of Open SVG 2003, 2003

[20] Chang, Yi-Hong & Chuang, Tyng-Ruey . *Online Aggregation and Visualization of Census Data: Population Mapping with SVG, XML, and Free Software.* In: Proceedings of SVGOpen 2002, Zurich, Switzerland.

[21] Lewis, C.T., Karcz, S., Sharpe, A., Parki, I.A.P.*BioViz: Genome Viewer. Development of an SVG GUI for the visualization of genome data.* In: Proceedings of SVGOpen 2002, Zurich, Switzerland.

[22] *Chaitan Baru, Amit Behere, Charles Cowart ,* Representation and Display of Geospatial Information: A Comparison of ArcXML and SVG, Second International Conference on Web Information Systems Engineering (WISE'01) Volume 2, December 03 - 06, 2001

[23] Arei Kobayashi, Satoru Takagi, Naomi Inoue, Extensions of SVG for human navigation by cellular phone, Proceedings of the SIGGRAPH 2003 conference on Web graphics: in conjunction with the 30th annual conference on Computer graphics and interactive techniques , 2003

[24] Vector Graphics Format, LeadTools.com, http://www.leadtools.com/Imaging-Developer-Portal/sitemap26.htm

[25] Multimedia File Format Guide, http://www.rice.edu/fondren/erc/howto/formats.html.

[26] Xiaoyong Su, B.S. Prabhu, Chi-Cheng Chu, Rajit Gadh, Middleware for Multimedia Mobile Collaborative System, third annual wireless telecommunications symposium (wts 2004), may 14-15, 2004, CalPoly Pomona, Pomona, California, USA.

[27] Rynson W.H. Lau, Frederick Li, Tosiyasu L. Kunii, Baining Guo, Bo Zhang, Nadia Magnenat-Thalmann, Sumedha Kshirsagar, Daniel Thalmann and Mario Gutierrez, Emerging Web Graphics Standards and Technologies, IEEE Computer Graphics and Applications, Jan/Feb, 2003, pp 66-75.

[28] Binh Pham and On Wong, Handheld Devices for Applications Using Dynamic Multimedia Data, Proc. of the 2nd Intl. Conf. on Computer graphics and interactive techniques in Australasia and South East Asia, June 15-18, 2004, Singapore, pp 123-130.

[29] Sangmi Lee, Geoffrey Fox, Sunghoon Ko, Minjun Wang and Xiaohong Qiu, Ubiquitous Access for Collaborative Information System using SVG, Proc. of SVG Open / Carto.net Developers Conference, Zurich, Switzerland - July 15-17, 2002

[30] R. Rosenbaum, H. Schumann, and Ch. Tominski, Presenting Large Graphical Contents on Mobile Devices – Performance Issues, in: P.C. Deans (ed): E-Commerce and M-Commerce Technologies, IRM Press, Hershey, London, 2004.

[31] Cascading Style Sheets, http://www.w3.org/Style/CSS/

[32] Document Object Model, http://www.w3.org/DOM/

[33] Composite Capabilities/Preferences Profile, http://www.w3.org/Mobile/CCPP/