Project SILA Technical Design Document

Frank Yung-Font Tang, ytang0648@aol.com,
2003-07-11

Introduction

SILA is a version of Mozilla that is capable of rendering text using SIL's Graphite rendering technology. In every other aspect this software is a standards-compliant program and behaves similarly to other versions of Mozilla, including the ability to display standard HTML and XHTML with CSS stylesheets.

Purpose of This Document

This document defines the goal, requirements, and dependency of project SILA. This document also discusses the high level architecture of project SILA after a high level overview of the Mozilla text processing and display subsystem.

Project Goal

To enable minority language communities to publish on the Internet by integrating the Mozilla Internet application suite (Browser, Email , Newsgroups, HTML Composer, Address Book, IRC Chat program) with SIL's Graphite technology in order to display complex writing systems correctly.

Project Requirements

Platform Requirements: Microsoft Windows 2000 or Microsoft Windows XP.

Functional Requirements:

1. Complex writing systems display in
    a. Browser
    b. Composer
    c. Email
    d. Newsgroups
    e. Address Book
    f. IRC Chat
2. Text display
3. Focus on supporting what Graphite supports
4. Left-to-right writing systems (Note: Mozilla already supports some right-to-left writing systems such as Hebrew and Arabic)

Language Requirements: The following language data will be used to test SILA

1. Burmese
2. Burkina Faso (Latin script)

Mozilla Requirements:

1. CPU: 233MHz
2. 64 MB RAM
3. 27 MB of free hard disk space

Due to the limit of project time, we decided the following are BEYOND THE SCOPE of this project. We may work on the following issues in future releases.

- Different Operating Systems than Microsoft Windows 2000 and XP.
- Right-to-left writing systems (Note: Mozilla already supports Hebrew & Arabic without Graphite.)
- Text keyboard input
- Text selection and cursor movement

Project Dependency

This project is heavily dependent on the following software technologies still under development:

- *Mozilla* - the open source Internet application suite project (www.mozilla.org)
- *Graphite* - the open source complex writing system rendering engine (graphite.sil.org)

In additional, this project also depends on the following stable software technology:

- *COM* - the component technology on Microsoft Windows. We will use COM to integrate the two technologies mentioned above.

Design

This section discusses how we architected project SILA. Since Mozilla is the foundation of this project, we need first to review the Mozilla components which handle text processing and display. We will also briefly review the key features provided by silgraphite. Finally, we will outline the high level architecture of SILA.

Mozilla Text Processing Subsystems Overview

The following components in Mozilla process text information:

1. The network library, Necko, receives byte streams in TCP/IP packets, block by block.

2. Necko passes blocks of the byte stream to the HTML parser, XML parser, JavaScript interpreter, or CSS parser, depending on the MIME type

3. The HTML parser, XML parser, JavaScript interpreter or CSS parser

- Finds out the text encoding (charset) of the incoming data from meta data, attached label or other sources,
- Calls appropriate Unicode converter to convert the byte stream into a Unicode text data stream,
- Parses the Unicode text data stream into tokens according to the (markup or programming) language lexicon, and then,
- The HTML content sink builds up content model according to the (markup or programming) language syntax.

4. The view system invokes the layout subsystem reflow process to build up the frame model. It

- Reads text data from the content model,
- Calls the “line breaker” to find out logical (purely dependent on the Unicode text data) line break opportunity,
- Reads style information from the style model,
- Calls the platform specific GFX (Graphics) subsystem to measure the width of each line break opportunity to decide where to break the line, the size of each frame, and the position of each frame,
- Calls the platform specific GFX (Graphics) subsystem to display the text according to the associated style and given position.

5. The Windows specific GFX (Graphics) subsystem is a very powerful and complex subsystem. Basically, the interface provides three main features for the purpose of text display:

- Creates a virtual font with information from CSS or user preferences
- Measures the size of a Unicode string by giving a virtual font
- Renders the Unicode string by giving a virtual font and screen coordinate

The Design, Features and Architecture of Mozilla Text Display Subsystems

Since the goal of the SILA project is to render the text of complex writing systems on Microsoft Windows, most (if not all) of the integration points reside in the Windows specific GFX (Graphics) subsystem. We review the design of this component in detail in this section. The ultimate text display code is inside the Windows specific GFX subsystem. We will only discuss the part which is related to virtual fonts, text measuring and text rendering.

Each virtual font in the Mozilla GFX subsystem has a defined order to access (if necessary) all the installed fonts to measure or render a given Unicode string. The order depends on:

1. The given CSS font-family property, or HTML `<font>` face attribute of that node or its inherited value
2. The per-language group font preference value, selecting by the source charset (text encoding) of the document, and the CSS generic font-family name
3. Other font preference values in the same language group
4. Font preference value of other language groups
5. The rest of the installed fonts

When the subsystem performs Unicode text measurement and rendering, it will first use the highest priority font to render all the text. It examines the 'CMAP' of each font to decide which Unicode characters could be measured / rendered by the first font. If the 'CMAP' of such font does not have a valid slot for such Unicode code point, then the GFX will load the next font in the order to measure/render those particular characters which cannot be rendered by the first font. GFX repeats this process through the list until all the Unicode code points could be rendered or it examines every single installed font.

In summary, the real font(s) used by the Mozilla for a given text node is mainly controlled by the following three factors:

1. The font-family property in CSS
2. The Unicode code points
3. The 'CMAP' table of the font

Key Features provided by Graphite

Graphite helps Windows-based applications to render text in complex writing systems by reading data from Graphite-enabled TTF fonts.

Graphite reads the additional Graphite-specific TrueType table and analyzes the given Unicode text to perform text measuring and rendering of complex writing systems. Please see the Graphite documentation for details. We will not discuss them in this section.

In order to work with Graphite, the calling software needs to implement two helper classes. The first one implements the IGrTextSrc interface to maintain the text data. The second class implements the IGrGraphics interface to read the TrueType font table and call the operating system to measure / render the "glyphs" (not the "characters").

The calling software should create a GrEngine for a specific font. It should associate an IGrGraphics to the GrEngine when initializing the GrEngine. It then passes the Unicode string to the FindBreakPoint method of the GrEngine object to create one or more IGrSegment. Then it calls the measurement and rendering method of the IGrSegment. These methods will call the methods implemented in the IGrGraphics to interact with the operating system.

High Level Architecture of SILA

In order to integrate Graphite into Mozilla, we need to address the following architecture issues first:

1. When should Mozilla call Graphite

2. How should Mozilla call Graphite

When to call Graphite

To answer the first question, we first need to look into the Graphite-enabled font. A Graphite-enabled font is really just a standard TrueType font (possible also OpenType or AAT) with an additional Graphite-specific table. The 'CMAP' of the Graphite-enabled font follows the same rule as a standard TrueType font. Therefore, we can share the same 'CMAP' exams code with other TrueType fonts. Mozilla should call the silgraphite only when it tries to use the Graphite-enabled font to measure or render Unicode text which is covered by it.

Since every Graphite-enabled font includes an 'Silf' table, we can distinguish a Graphite-enabled font by checking the existence of this table in the TrueType font directory. If a font includes such TrueType table, then we will call silgraphite to perform the text measurement or rendering.

How to call Graphite

To reduce the maintainance cost and reduce the need to use most of the code across different versions of Mozilla, we defined a generic COM-based interface to separate the Graphite-specific code from the Mozilla-specific code. The Graphite-specific part is implemented in a COM-based DLL, currently named mg2.dll (Mozilla Graphite Together). This DLL includes a statically linked Graphite library of a stable version. It hides all the details of how to work with the Graphite APIs. The Mozilla code will be changed to call this generic COM interface. The Mozilla patch implements all the details of how to work under the Mozilla GFX virtual font look up and text measurement/rendering subsystem.

Project SILA

1. ISpecialTTFFont.h: A COM interface definition which measures and renders text

2. Mozilla

   - GFX
     - Patch to call the ISpecialTTFFont COM interface when measuring/rendering text if it is a Graphite-enabled font
   - Xpinstall
     - Patch to include mg2.dll in the installer, and install the mg2.dll, which includes register mg2.dll to the registry for COM, in the installation process

3. Mg2.dll

   - silgraphite: A stable version of Graphite library statically linked into this DLL
   - silFont: Class which implements the defined ISpecialTTFFont COM interface by using GrGraphics, GrTextSrc and calling silgraphite
   - Registry: Code to perform COM DLL self registry
   - GrTextSrc: Helper class implements IGrTextSrc
   - GrGraphics: Helper class implements IGrGraphics

How to Generate Web Content to Work Nicely with SILA

To make content (HTML) work nicely with SILA, we need to:

1. Clearly specify the font name in the CSS font-family property; this is especially important when working with Latin-based Graphite-enabled fonts to avoid the automatic font substitution mechanism in Mozilla which may substitute non-Graphite-enabled fonts and thereby interfere with Graphite rendering.

2. Use correct and appropriate Unicode code points. Either use UTF-8 to encode the document, or use text encoding neutral escaping mechanism in each language to specify the Unicode code point.

Please refer to the CSS1, CSS2 and HTML 4.01 specification from w3c.org site for details.

Project Risks & Solution

Stability of Mozilla: Since this project heavily depends on Mozilla, the status of the Mozilla project is the biggest risk of this project. To address this, we decided the following:

1. Use the Mozilla 1.0 branch (which Netscape 7, Netscape 7.01, Netscape 7.02 are based on) as the base to produce the Alpha build

2. Use the Mozilla 1.4 branch (which Netscape 7.1 is based on) as the base to produce the Beta 2 build.

Stability of Graphite: The bugs and stability of Graphite also greatly impact this project. In order to reduce the risk, we decided to statically link the Graphite library into the mg2.dll. The version of Graphite will depend on the addressed issues in the Graphite source, number of changes around the date, and trial testing result.

Project Release Plan

Based on the status of the project, we will release the binary in the following format:

1. Patched gkgfxwin.dll + mg.dll in zip file: We will release Alpha build by including a modified gkgfxwin.dll based on the same branch of Netscape 7.0 (or Mozilla 1.0) and mg2.dll. In order to use it, the user has to first download the Netscape 7.0 (or Mozilla 1.0) separately, download the package to replace the gkgfxwin.dll manually, and call the COM registry program that comes with Windows to register the mg2.dll.

2. Mozilla files + mg2.dll + setup.bat in zip file: We will release in this way in the Beta cycle until we finish the required installer changes. This requires the user to download the patched Mozilla in a zip file and run a setup.bat batch program after unzip the zip file to install.

3. Mozilla installer includes everything: This is the ideal way to release Project SILA. We should ship Beta 2 in this method. This requires installer changes in addition to the run time GFX, silgraphite integration. The user will use the installer exactly the same way they use the Mozilla installer.

Installation

The installation program is called sila20030625inst.exe. It requires 27 megabytes of disk space. To install Mozilla, download the installation program[1], double click on the file and follow the instructions.

Using Graphite fonts

To use a Graphite-enabled font within Mozilla, simply install the font into your Fonts control panel in the normal way. Any text that is marked up to use that font will use the Graphite tables to perform the rendering.

Click here to download sample Graphite fonts[2].

[1] http://sila.mozdev.org/silab2.html

Links

- SILA[3]
- Mozilla and Netscape family of products
    - Mozilla[4]
    - Netscape[5]
- Graphite
    - Graphite[6]
    - Download Graphite fonts[7]
- Standards that are used for creation of web-based materials
    - XHTML[8]
    - CSS[9]

This document was generated from
http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&item_id=Mozilla_GraphiteDoc
on Fri, 29 Aug 2003 08:11:17 -0500.

END

[2] http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&item_id=GraphiteFonts

[3] http://sila.mozdev.org

[4] http://www.mozilla.org

[5] http://www.netscape.com

[6] http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&item_id=CatGraphite

[7] http://scripts.sil.org/cms/scripts/page.php?site_id=nrsi&item_id=GraphiteFonts

[8] http://www.w3.org/TR/xhtml1

[9] http://www.w3.org/Style/CSS