Dynamic Charting Framework for graphical analysis in Trace and Debug Tool

By

Shishir Sompura

13MCEN23

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AHMEDABAD-382481

May 2015

Dynamic Charting Framework for graphical analysis in Trace and Debug Tool

Major Project

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in Computer Science and Engineering(Network Technology)

By

Shishir Sompura

(13MCEN23)

Guided By

Prof. Monika Shah

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AHMEDABAD-382481

May 2015

Undertaking for Originality of the Work

I, **Shishir Sompura**, Roll. No.**13MCEN23**, give undertaking that the Major Project entitled **"Dynamic Charting Framework for graphical analysis in Trace and Debug Tool**" submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in **Computer Science and Engineering(NT)** of Nirma University, Ahmedabad, is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

Signature of Student
Shishir Sompura - 13MCEN23

Endorsed by:

Prof. Monika Shah

Certificate

This is to certify that the Major Project entitled **"Dynamic Charting Framework for graphical analysis in Trace and Debug Tool** " submitted by **Shishir Sompura (13MCEN23)**, towards the partial fulfillment of the requirements for the degree of Master of Technology in **Computer Science and Engineering(NT)** of **Institute of Technology, Nirma University, Ahmedabad** is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of my knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

Mr. Vinod Allu Sivala
Team Lead,
Intel Mobile Communication, Bangalore.

Prof. Monika Shah
Guide-Assistant Professor,
Nirma University, Ahmedabad.

Mrs. Ramandeep Kaur
Engineering Manager
Intel Mobile Communication, Bangalore.

Prof. Gaurang Raval
PG-Coordinator(CSE-NT),
Nirma University, Ahmedabad.

Dr. Sanjay Garg
HOD (CSE Dept.),
Nirma University, Ahmedabad.

Dr K Kotecha
Director,
Nirma University, Ahmedabad.

Abstract

Dynamic Charting Framework for graphical analysis in trace and debug tool contributes towards the success of organization in analyzing system behavior. Trace tool captures different traces from the hardware and software Component and these traces are used for analyzing system behavior. Manual process makes it very hard for the analyzer to analyze these traces. As the traces coming from component like LTE, 3G and 4G have a wide range of parameters (RSRP, RSRQ, UL PRB, DL RNTI etc.), The computation required for these parameters is very complex and needs human effort for calculating these parameters for a set of traces which in turn introduces delay in the analyzing process. To minimize these difficulty and delay, Dynamic Charting framework will be implemented reducing the human intervention making the analyzing process more interactive and easily understandable.

Acknowledgements

First and foremost, sincere thanks to **Mrs. Ramandeep Kaur**, Manager,Platform Tool Development, Intel Mobile Communication India Private Limited, Bangalore. I enjoyed her vast knowledge and owe her lots of gratitude for having a profound impact on this report.

I would like to thank my Mentor, **Mr. Vinod All Sivala**,Team Lead, Intel Mobile Communication India Private Limited, Bangalore for her valuable guidance. Throughout the training, She has given me much valuable advice on project work. Without her, this project work would never be completed.

My deepest thanks to **Prof. Monika Shah**, Assistant Professor, Department of Computer Science and Engineering, Institute of Technology, Nirma University, for giving me an opportunity and guidance throughout the project. It was only due to her valuable opinion, cheerful enthusiasm and ever friendly nature that I was able to do part of my research work in a respectable manner.

It gives me an immense pleasure to thank **Dr. Sanjay Garg**, Hon'ble Head of Computer Science and Engineering Department, Institute of Technology, Nirma University, Ahmedabad for his kind support and providing basic infrastructure and healthy research environment.

Last, but not the least, no words are enough to acknowledge constant support and sacrifices of my family members because of whom I am able to complete the first part of my dissertation work successfully.

Shishir Sompura
13MCEN23

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

Trace and Debug tool maintains the log of traces which are captured from the User Equipment. User can analyze the trace using this tool.

1.2 Objective of Study

Charting in Trace and Debug tool is very useful for the user to analyze the User Equipment behavior. As Manual process makes it very hard for the analyzer to analyze the hardware traces, So Charting framework will be implemented reducing the human intervention making the analyzing process more interactive and easily understandable.

1.3 Requirement of Dynamic Charting Framework

As the evolution of telecommunication, new technologies like 3g and 4g has evolved with high speed data transfer, low latency and packet optimized radio access technology supporting flexible bandwidth deployment. Same times its network architecture has been designed with the goal to support packet switched traffic with seamless mobility and great quality of service. In LTE or any other cellular network, User Equipement(UE) report some sort of signal to base station for various decision making. It can be used for down link scheduling(using CSI), cell selection, Up link scheduling, Hand Over, cell reselection, calculation of uplink and downlink path loss for power control, multipath propagation, uplink interface and for location base service. All of these are achieved by different parameter called RSRP, RSSI, RSRQ, and RSTD. As this parameter has a wide range of values, we can create plugin by using charting framework but it would be very difficult because there are various parameters So Its not good to have a dedicated plugin for creating a chart for one parameter. And another cause is SWT charts is not as powerful plotting tool as comparison to the other available. So there is something require that can overcome this problems.

Chapter 2

Literature Survey

This chapter provides an overview of the research done in the field of System Software. Here the first section defines why the trace is important. What is the old concept of tracing and advantage of new concept of tracing and then the architecture of System trace tool.Packet Header defined for the tool.

2.1 Background

This section provides introduction to intel in mobile platform.

2.1.1 Cellular Modem

Mobile modem platform consists of processors and other peripherals for example:- USB, Wi-Fi, Bluetooth, Battery power and etc. The processors are communication processor and application processor. Mostly peripherals contain connectivity and data storage contents in Modem Hardware. [2]

Modem generates various type of Signals this are 2G, 3G, 2g FW (2g firmware), 3G FW,4G, etc.[2] Modem cores produce this signals for each events. These signals contain the packets for the particular type.

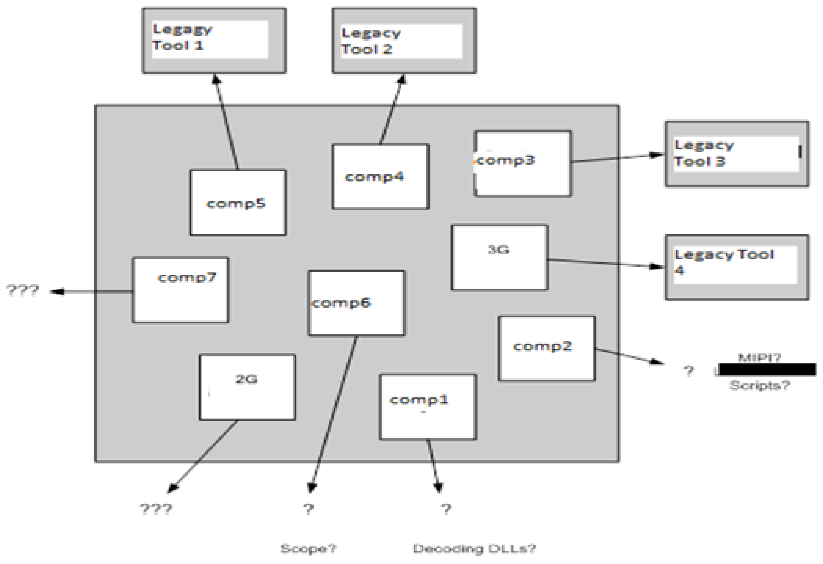

Figure 2.1: Tracing from different component(Reprinted from [2])

2.1.2 Importance of tracing

Tracing of a software and hardware behavior is a very crucial task. With this traces we can analyze the system behavior and find out issues during development and production. It is essential during conformance, interoperability.[2] Lets take an example for describing where the tracing is needed for the modem analyzer. When user is making a call using the mobile and is moving from one cell to another one (hand-off) then in between the mobile call gets disconnected due to some reason, then there may be some chance of crash happen in the modem. So if we are continuously capturing traces from the modem, then by analyzing the log, we can come to know the reason behind this crash (core dump).

2.2 Overview of System tracing

Traces are generated by the software running on the cores.[2] These cores allow tracing of mobile software behavior and its interaction with the network.[2] For testing the functionality, these traces are very useful. The trace information can be transferred to the outside world over one of the standard communication interfaces like: USIB, USIF, MIPI etc.

There are different component that generates traces that requires high bandwidth trace stream and some requires low bandwidth trace stream.[2] Each component generates a signal and is transferred to X which is a hardware that is configured to choose traces defined by AT Commands. Then this hardware transfer these traces to trace tool.[2] For capturing traces, we have two ways: -

- For high bandwidth traces, we have dedicated stream that transfers the high bandwidth traces to Trace Box. The Trace Box contains the storage device where we can store that trace and later we can analyze them.[2]

- For low bandwidth trace, We have a stream like USB and U-ART. That will transfer the traces directly to the trace tool.

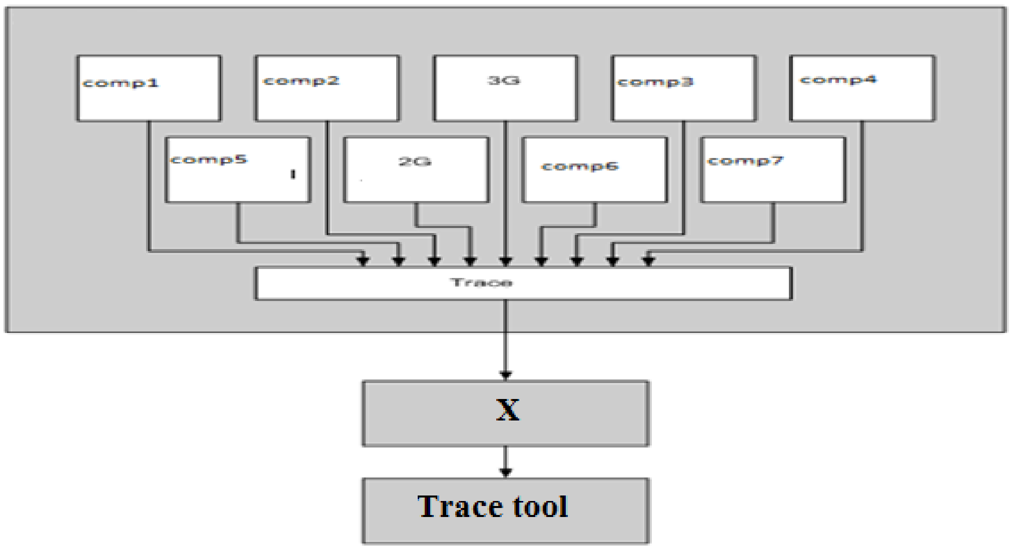

Figure 2.2: Trace Concept (Reprinted from [2])

2.3 Trace tool Setup

[2] The system architecture for tracing contains different component:-

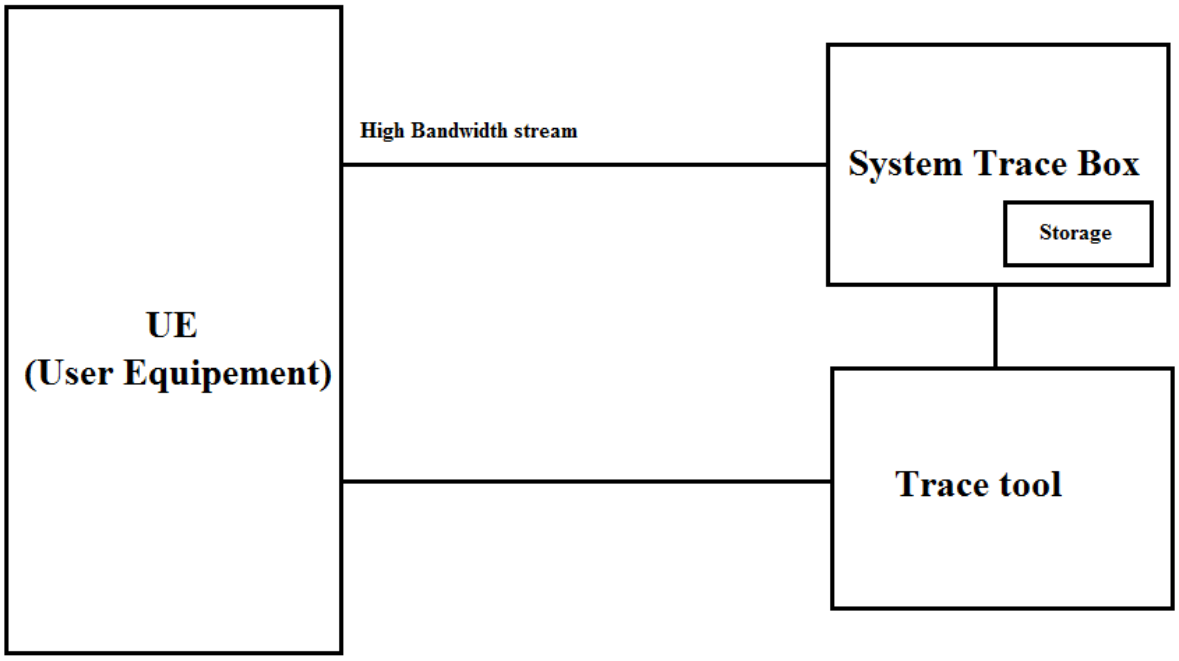

Figure 2.3: Trace Tool Setup

- **User Equipment**: - It could be either mobile or a test hardware setup. User Equipment contains the modem for capturing traces. The hardware Setup contains communication port (high bandwidth communication stream port and low bandwidth communication stream port).
- **Trace Box**: - It has a storage device which can store some traces. Trace box stores the traces that are coming from high bandwidth stream. In trace box, we can also filter the traces according to their unique identifier.[2]
- **Trace Tool**: - Trace tool records and decode these traces and provides an easy graphical interface to view, search and analyze trace data. Trace tool can run in two modes: - Headless Mode and UI mode. Headless Mode is just a command line interface from where we can trigger different operation via firing a specific

command. In UI Mode, We have a User Interface, in that User have different Option for performing specific operations.[2]

2.4 Trace Tool Architecture

The architecture of trace tool is shown below: -

Architecture.png

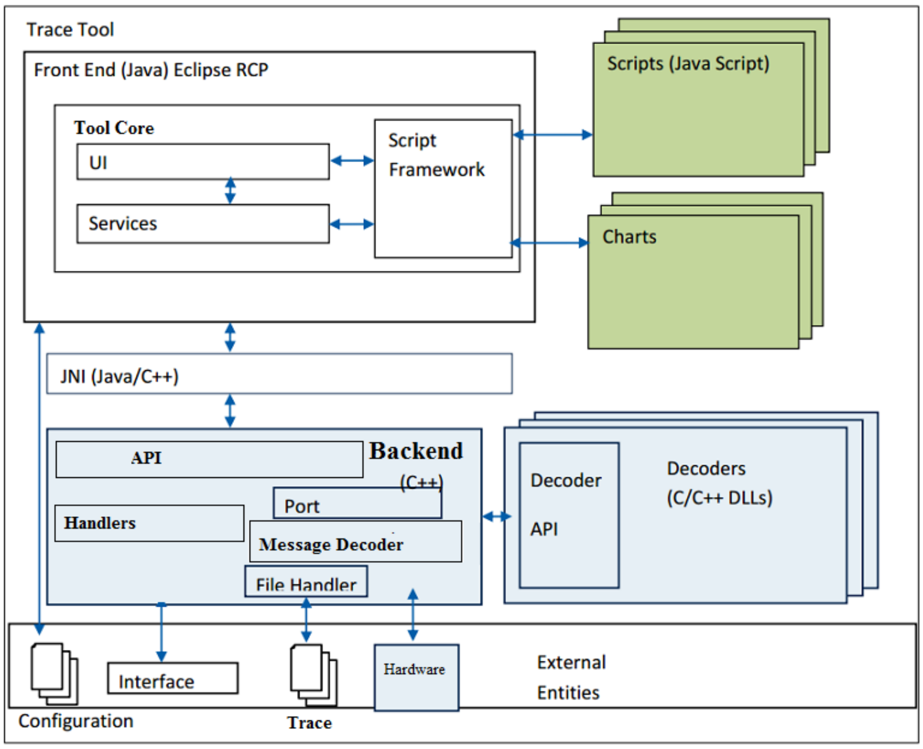

Figure 2.4: Trace Tool Architecture (Reprinted from [2])

2.4.1 Trace Tool Front End

Trace tool Front End is developed in java with Eclipse RCP Plugin development. Eclipse provides the plugin environment as well as platform to work on[4] It is developed in Eclipse IDE (Integrated Development Environment). IDE is defined as a collection of tools and platforms. Where tools are plugins and platform refers to our java platform. Eclipse is an IDE because it provides plugin development as well as platform (RCP). Now the eclipse also gives us some API that helps us in creating a

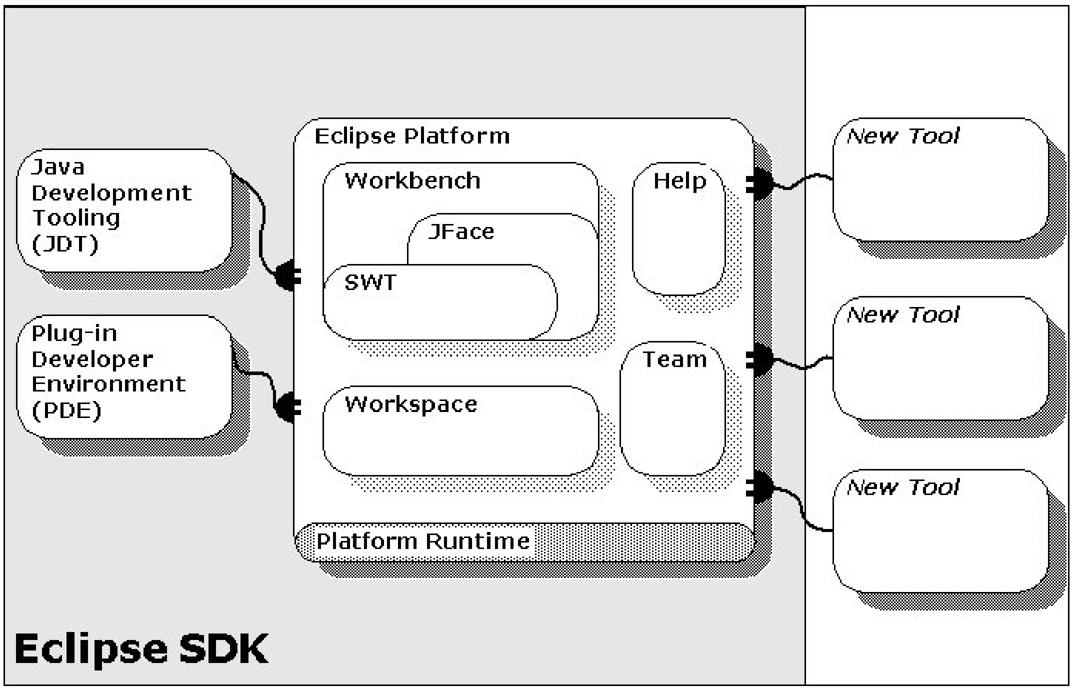

Figure 2.5: Eclipse Platform Component(Reprinted from [6])

good UI component. Eclipse Architecture is shown in below figure5. It contains five components:-

- **Java Development Tooling**:- JDT provides a set of plug-ins that add the capabilities of a full- featured java IDE to Eclipse platform.[6] The JDT plugins provides APIs so that they can themselves be further extended by other tool plugin builder. The JDT plugins are categorized into:
  - **JDT APT**
  - **JDT Core**
  - **JDT Debug**
  - **JDT UI**
  - **JDT Text**

- **Plugin Development Environment**[6]:-The plugin Development environment(PDE) provides tools to create, develop, test, debug, build and deploy Eclipse plug-ins, features, updates sites and RCP products. PDE also provides OSGI tooling, which makes it an ideal environment for component programming, not just Eclipse PDE. There are three component of Eclipse PDE:- Build, UI and API Tools. Each of these components operates like a project unto its own, with its own set of committers, bug categories and mailing list.

- **Eclipse Platform**:-Eclipse platform contains four components: - Workbench, Workspace, Help and Team.[6] Eclipse Platform contains several of API like SWT and JFACE. As we know in java we have swing and AWT framework but this are OS dependent means they use only OS specific item for creating GUI component. So Sometimes if we dont want to use the OS specific items then Eclipse has created its own GUI component and created some APIs for them. So whenever we want to create our own GUI component then we can use them.

- **Platform Runtime**

- **New tool/Plugins**

2.4.2 Trace tool Backend

Backend is used to capture and decode trace messages.[2] The Backend contains the code for recording traces and interfacing to the Decoders and is implemented in C/C++ (as this was shown to be able to process data approximately twice as fast as Java). A consequence of the language split is the need for all decoded data to be passed upwards through the JNI to the Front End. Due to the predicted JNI bottleneck, the API to the Backend was designed to minimize to a practical extent the amount of data that needs to be passed through the JNI (for example by only returning the needed parts of needed messages, and only upon request rather than as a continuous stream). The Backend was also designed so that it could be used stand-alone with a different front end (although this was not a requirement and was only used as a guideline to help enforce a clean and well documented interface).

Backend contains Backend API, handlers that interacts with different interfaces. Backend interacts with the Hardware and the Decoders and its API.Backend also maintains the Message database for storing the traces. Message database provides easy writing and reading of messages, possibly using several files within the session. In this, the recording speed is currently only limited but OS access to the mass storage device. The Databases class stores messages in the order that are written to it, and indexes them in this order. There is a possibility that there will be a future requirement to be able to reorder the messages in the Database according to GTS. If this changes in the future, a new version of the Database could be written which provides extended indexing information based on GTS to make such reordering efficient.

2.4.3 Decoders

Decoders in trace tool are used for decoding the trace message.[2] There is one YDecoder class processes all incoming Y streams. It decodes Y data link and transport layers to and converts to a series of objects type Messages. This messages are then stored in database.

Trace Messages are primarily decoded by Decoders which are developed and maintained external to trace tool. Trace tool defines an interface which these decoders must adhere to, and it is defined in a header file.[2] Decoder is a single DLL with no extra file. After decoding, Each message is decoded into several data fields. The name and type of the field depends upon the message unique identifier, and are defined by the decoders.

The Backend decodes one Trace Message at a time. This message is always specified by a search of the message database, starting from a defined index and in a defined direction.[2] Typically the user of the Backend will want to decode a range of messages; this is done by using a series of searches, with each search starting at the position the previous search ended. The Backend always returns to the caller the position a search ends to make this algorithm easy for the caller to implement.

For the special case of live capture, where it is desired to always display the most recent messages, it is recommended to search the message database backwards from the end-of-file. In this way the most recent trace messages will be displayed, even if the trace data rate is faster than the caller can keep up with.[2] The Trace Tool is designed for the analysis of Trace Messages. It is presumed that he user of the Trace Tool knows what sources of Trace Messages he is interested in. These sources are specified to the Backend in terms of a unique identifier list. With each request for finding and decoding a message, the list is passed and used to filter which messages are found by the search and decoded.

2.4.4 Scripts Framework

The scripting framework is implemented using Java 1.6 scripting framework. It supports Java scripts. Script functionalities include: -

- Post process the messages
- Operate the STT in an automated environment

When more complex fields are required on a message, which cannot be provided by the standard decoder, a script can be used. Script offers almost all the functionality, which the GUI provides a user. More functionality will be added to the set of APIs based on the requirements.Scripting framework provides standard interfaces, which it requires the implementation from every script to run. A Trace Tool script should implement the standard mandatory functions that the scripting FW expects for a successful script execution.

- **Script LifeCycle**

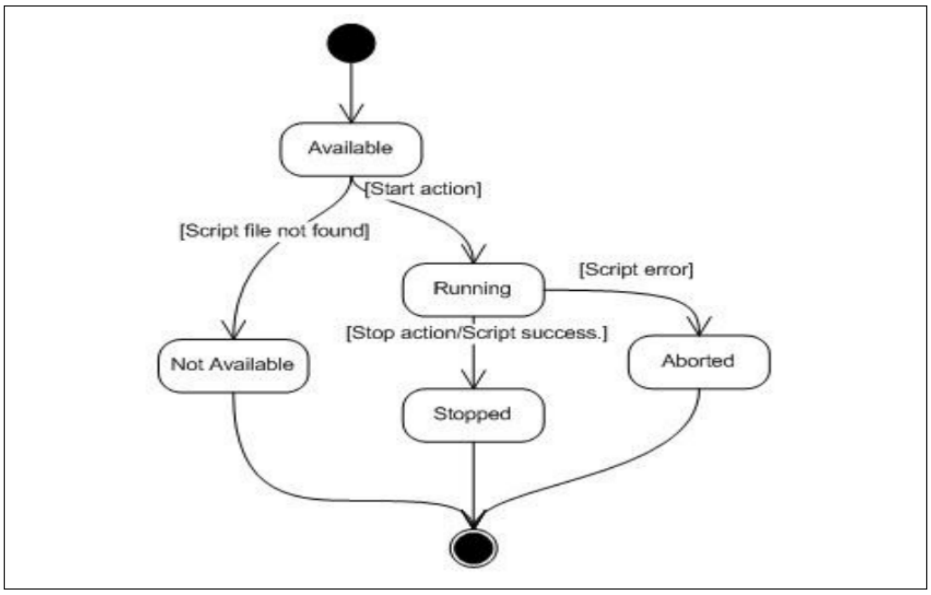

Figure 2.6: Script LifeCycle (Reprinted from [2])

2.4.5 Charting Framework

Chart base framework (independent of any network technology) supplied by Trace and debug tool how to set up the environment to create plug-in based charts and create your own framework enhancements. This framework contains general functionality for rendering the chart but no logic on how to calculate actual data. Additionally, some functionality to handle environment independent data is provided. In this charting framework, it uses the SWT chart for plotting.

Chapter 3

Problem Statement

3.1 Problem Statement

Dynamic Charting Framework for graphical analysis in Trace and Debug Tool

3.2 Significance

Trace Tool captures trace from the User Equipment and maintains these traces in a Y file.For large set of traces, User usually tries to create a graph for easily analyzing the behavior. For large set of data it is very difficult to maintain the data and plot it. So there should be some functionality added to the Trace Tool that will create a required Graph/Chart for the User.

Chapter 4

Methodology for Implementation

4.1 Objective

Dynamic Charting for graphical analysis in Trace and Debug Tool. As discussed in the previous section, trace tool should have some mechanism to plot a chart for network analysis. So that User will be able to create throughput charts, RSSI/total gain charts, cell measurement charts and etc. But as the evolution in telecommunication technology, there are different protocol evolved like LTE, 4g and 3g. In these cellular network, UE reports some sort of signal to base station for various decision making. These signals can be used for down link scheduling(using CSI), cell selection, Up link scheduling, Hand Over, cell reselection, calculation of uplink and downlink path loss for power control, multipath propagation, uplink interface and for location base service. These signals have wide range of parameters like RSRP,RSRQ,RSSI,RSTD and etc. If User wants to analyze the above characteristics then he needs to compute these parameters. But as the large amount of traces and wide range of these parameters, It would be very difficult for the user to analyze the network behavior. So Dynamic Charting will help user by integrating these charting concept with the trace tool and simplify the analysis process for the user.

4.2 Design Approach

4.2.1 Design Approach 1

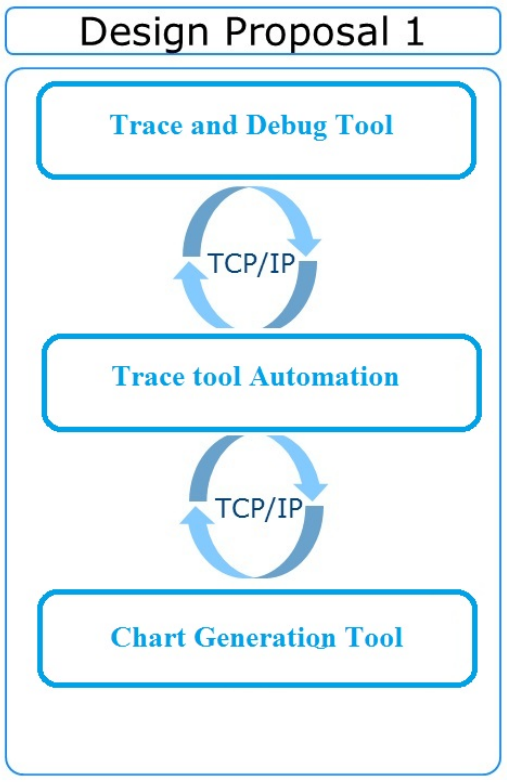

Figure 4.1: Design Proposal 1

- **Pros :-**
  - Reuse of logic and no code duplication
  - Support Command line interface
  - Plot resource will be attached with decoder, So GUI will be dynamic and no need to load all the resources and manage all the plot resources separately.
- **Cons :-**

- Data sync with the trace and debug tool and Chart Generation Tool plot will be lost (because of the difference in the data processing rate)
- The two processes uses different way for processing the messages. This can result in inconsistency in the data presented in trace tool and in chart.

4.2.2 Design Approach 2

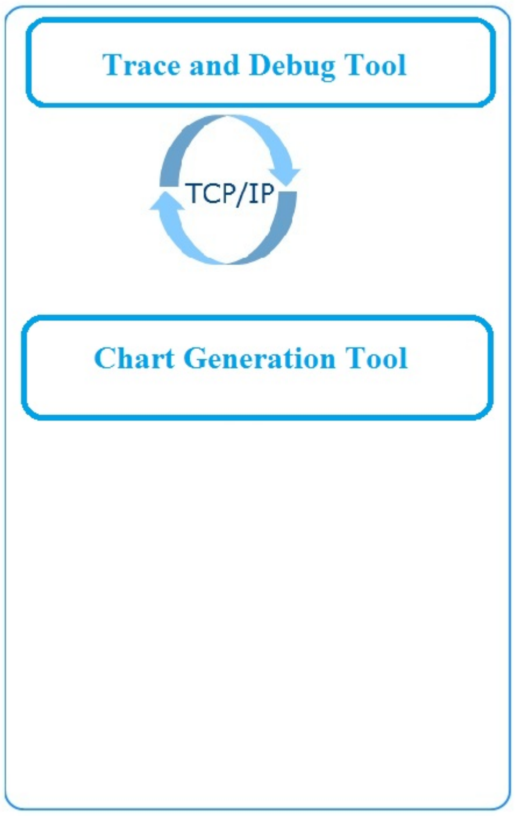

Figure 4.2: Design Proposal 2

- **Pros :-**
  - Due to recent changes in Trace and Debug Tool (Engine pre-filtering) time to plot the resource is much faster. Chart Generation tool would also have the same benefits.

  - Using TAV fallback mechanism Trace and Debug Tool can support all the plots that are delivered with CAPS

- **Cons :-**
  - Code duplication till we have final solution in place
  - Performance issue when plot resource is dependent on Structured Data

4.3 Approach to be implemented

Based on the pros and cons of the approaches, Design approach 1 is the most suitable for implementation, which is much more effective than the other two approaches. The advantage of this approach is that Code will not be duplicated and it will also support command line interface.

4.4 Challenges

So for implementing these Charts, There are different plotting tool available.

4.4.1 Challenges in Plotting

- Trace tool does not have any way to interact with Plotting Tool.
- There should be some way defined for creating a Plotting Tool script that should contain X-Y data points.
- As we know that in traces, same fields are not filled for each trace. So while plotting, we need to cache all of the field that will increase communication overhead.
- It's very difficult to manage plot resource. As there are lot of plot resource for each component, so it will consume more space and complex to manage all plot resource separately.

Chapter 5

Implementation

5.1 Implementation Methodology

It involves the implementation of the Trace Tool Automation that is responsible for interacting with trace tool and Chart generation tool. The below diagram shows the interaction between the trace tool and Trace Tool Automation. The whole process includes the below functions :-

- Launch Trace Tool Automation.exe.
- Create Socket Connection between trace tool and Trace Tool Automation.
- Create Socket Connection between Trace Tool Automation and Chart generation tool.
- Create Session and based on the decoder loaded generate an xml which will be sent back to trace tool that will be used for generating UI.
- Export the trace to a specific format and generate cached file.
- send plot resource to Chart generation tool.
- send the cached file to Chart Generation tool.
- Appropriate Chart will be generated.

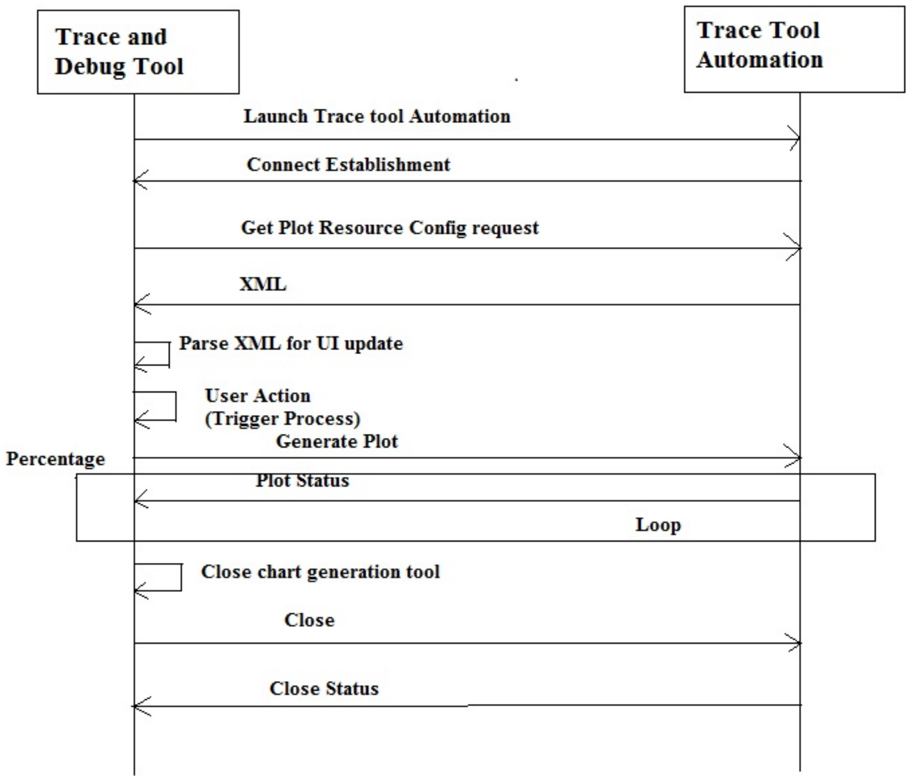

Figure 5.1: TraceTool Automation Automation and Trace tool Interation

5.1.1 Plot Resource

Plot resource is a file that is written in the format that chart generation tool can understand. This file contains the information about what to plot and how to read the cached file.

5.1.2 Trace Tool Automation

Trace Tool Automation is a tool which browses the high bandwidth trace messages and performs several actions, like script executions or pre-defined actions.

Trace Tool Automation is dedicated for automation. It is intended to be used in batch-files.Trace Tool Automation takes istp file and decoders and based on that will generate an xml file and send it back to STT for GUI update. It also sends the selected plot resource to Chart Generation tool and export the traces to different file format.

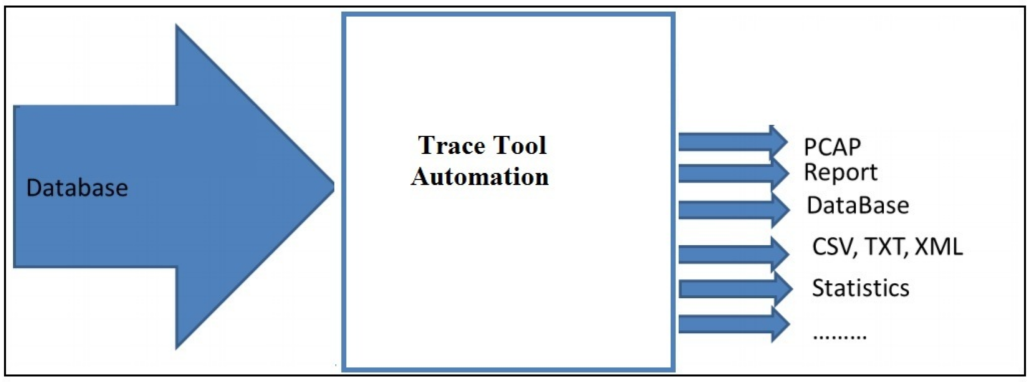

Figure 5.2: Trace Tool Automation

5.1.3 Cached File

Cached File contain the data what we want to plot. This file structure contains data followed by a header. These have solved one more challenge: - **As we know that in traces, Same fields are not filled for each trace. So while plotting, We need to cache all of the field that would require some effort in reading as tools needs to read field that doesn't have any data.** As it adds a header that contain the some unique identifier and field which are filled. It also prefixes this unique identifier and cache only filled fields. As this would increase quite some processing overhead but it will help in reducing communication overhead.

5.2 Plotting a chart

5.2.1 Genration of Chart

Chart Generation is done in following steps:-

- Initially for creating a chart, Tool launch Trace Tool Automation.exe. and start Server and create a TCP connection between Trace Tool Automation and Trace Tool.

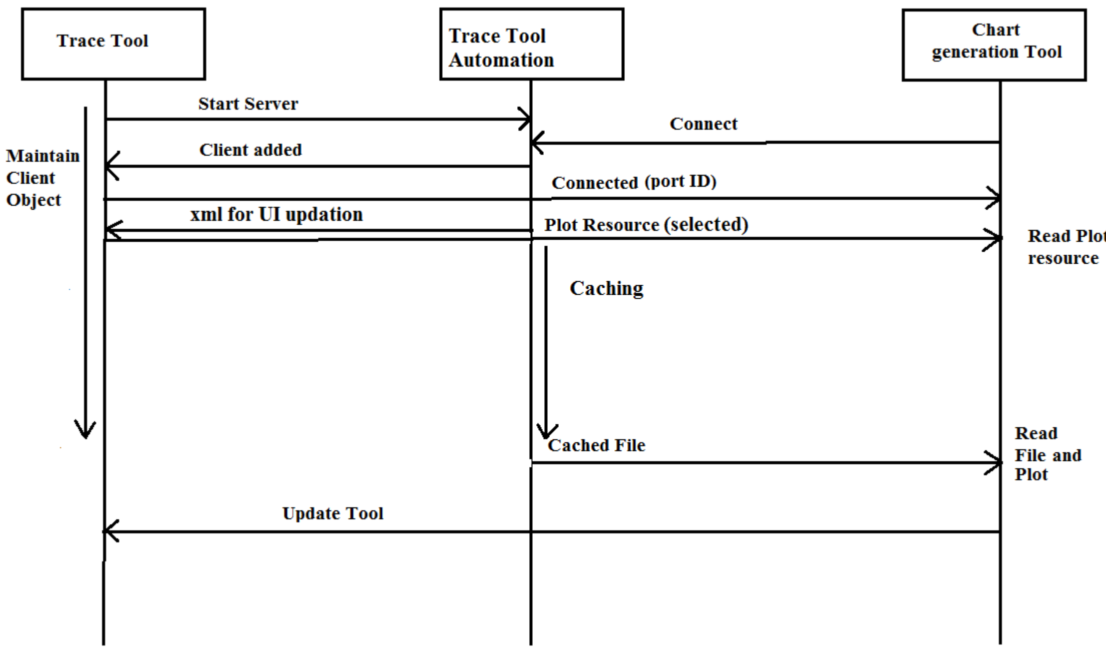

Figure 5.3: Handshake

- Then Trace Tool Automation generate the xml file by extracting resource file from decoder. And it sends the xml to trace tool for GUI update.
- User select the plot resource to be plotted and send Plot request to Trace Tool Automation, then it will create a TCP connection to Chart Generation tool and send plot resource.
- Trace Tool Automation also start generating the cache for the selected resource.
- Once it is done then send the cached file to Chart generation tool and plot will be generated.

5.3 Example of Plotting with LTE plot resources

Here is the example of plot generated in PLOTTING TOOL for LTE(long term evaluation) traces.So as we have discussed in the chapter 2 about LTE plot. Like there are different parameter that are used for calculating down link scheduling(using CSI), cell selection, Up link scheduling, Hand Over, cell reselection, calculation of uplink and downlink path loss for power control, multipath propagation, uplink interface and for location base service. So here i am plotting a chart of RSRP (Recieved signal received power) that gives some information about signal strength while hand offs.

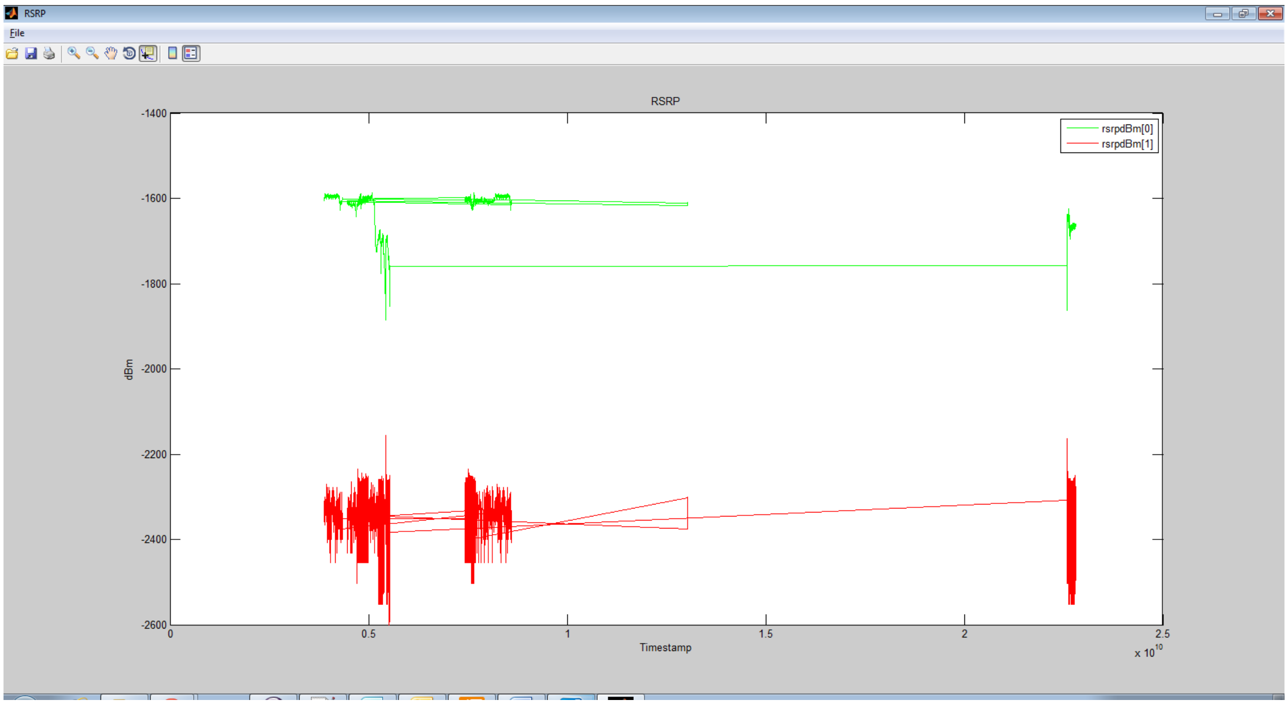

Figure 5.4: Plot of RSRP(LTE)

Chapter 6

Methodology Tools

6.1 Tools

6.1.1 GIT

GIT is a version controlling tool used for maintaining project.[7] It maintains repositories of the project and allows developer to work parallel on same file in repositories. And also allow some merging tool for conflicts. Developer can create more than one branch and do their own changes in it.[7]

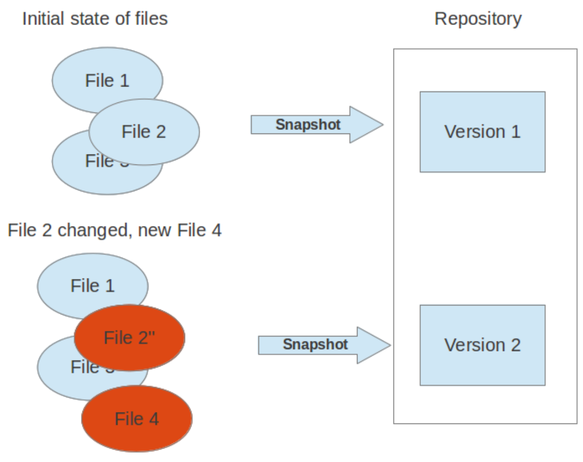

Figure 6.1: GIT

6.1.2 Beyond Compare (Licensed tool for comparison of text, folder, zip)

Beyond Compare is a merging tool which is used for merging files that have some conflicts.

6.1.3 Shared drives

In Shared Drives, we maintain the all release update and tools so that any developer could access to the tool and release whenever required.

6.1.4 RTC Project Planning

This is an IBM tool for maintaining the project states. This is used for tracking project states.[3]

6.1.5 Testing Devices , Accessories and Tools

- **User Equipment**: - User Equipment is integrated circuit that have modem inbuilt on it and some testing environment. It user equipment contains the architecture of device modem. It contains MIPI1, MIPI2, USBH and USB on it.

- **Test Machine**

- **UI Test Automation Framework**: - Provides functional and regression test automation for software applications and environments.

6.1.6 Klockwork

It is used for the semantic analysis of the source code.[9]

6.1.7 Jenkins PCQG and Sanity

Jenkins is used for build generation.[8] In Jenkins it also run unit test cases that ensure the code quality. Sanity machine runs some basic set of QTP test cases, this ensures the quality product.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

As a conclusion, I would like to say Dynamic Charting framework would help analyzer in describing the system behavior easily.With this approach, we reduces the resource file space by attaching it within decoders. These charts can also be easily shared among the organization for further inputs.It will provide a great user experience in analyzing the traces.

Bibliography

[1] WhitePaper :- Using RTC to Manage Multi-Partner Project Governance at National Grid

[2] Intel Confidential Document for trace tool

[3] intel.library.com - Intel docs for propriety tools.

[4] An Eclipse Plugin for the Automated Reverse-Engineering of Software Programs,Dugerdil, P. ; Dept. of Inf. Syst., HEG-Univ. of Appl. Sci., Geneva ; Kony, D. ; Belmonte, J. , 2009

[5] Jagadish K(worker of INTEL), "IPC: AP- CP Communication:" `http://jagsposts.blogspot.in/2012/06/ipc-ap-cp-communication.html`

[6] Eclipse Java Development IDE `https://eclipse.org/`

[7] GIT `http://git-scm.com/`

[8] Jenkins `http://jenkins-ci.org/`

[9] Klockwork `http://www.klocwork.com/`

[10] The impact of channel environment on the RSRP and RSRQ measurement of handover performance ;He Xian ; Broadband Commun. Network Lab., Beijing Univ. of Posts Telecommun., Beijing, China ; Wu Muqing ; Miao Jiansong ; Zhang Cunyi , 2011.

Appendices