**Design**

**On**

**Automated Patch Mechanism for MR Software Components**

**SUBMITTED BY**

**PRAVEEN K 1MS12CS075**



**M. S. Ramaiah Institute of Technology**

**(Autonomous Institute, Affiliated to VTU)**

**BANGALORE-560054**

**Department of Computer Science & Engineering**

***Under the guidance of***

|  |  |  |
| --- | --- | --- |
| **Sriranga Natha**  **System Architect, PIC ADI Magnetic Resonance**  **Philips Innovation Campus, Bangalore** | **&** | **Sini Anna Alex**  **Assistant Professor**  **Department of Computer Science & Engineering** |

1. **Introduction**
   1. Description of the organization of the document

[Section 1.2](#numberofmodules): Number of Modules- Total number of modules

[Section 1.3](#modulesdescription): Modules Description- Description of the operations performed by each module

[Section 1.4](#algorithmdesign): Algorithm Design- Provides an algorithm for automated patch mechanism

[Section 2](#architecturedesign): Architecture Design- Design of the entire automated patch software system. It provides information on the decomposition of the system into modules.

[Section 3](#dataflowdiagram): Data flow diagram- Shows the flow of data through the software system.

[Section 4](#references): References- List of books, URLs, tools consulted and used to design the document.

* 1. Number of Module: 4
  2. Modules description
* *Identify and build the dependent DLLs on the developer’s system:*

The application takes as argument a list of modified files and the patch location. It finds the DLLs that use the modified file and builds them.

* *Identify and stop the dependent processes on the remote test system:*

The processes that need to be patched are identified and only these processes are stopped so that the patching can be done on them.

* *Target all the DLLs to their respective locations on the remote test system:*

The built DLLs are copied to their respective locations on the remote test system.

* *Start only the impacted processes:*

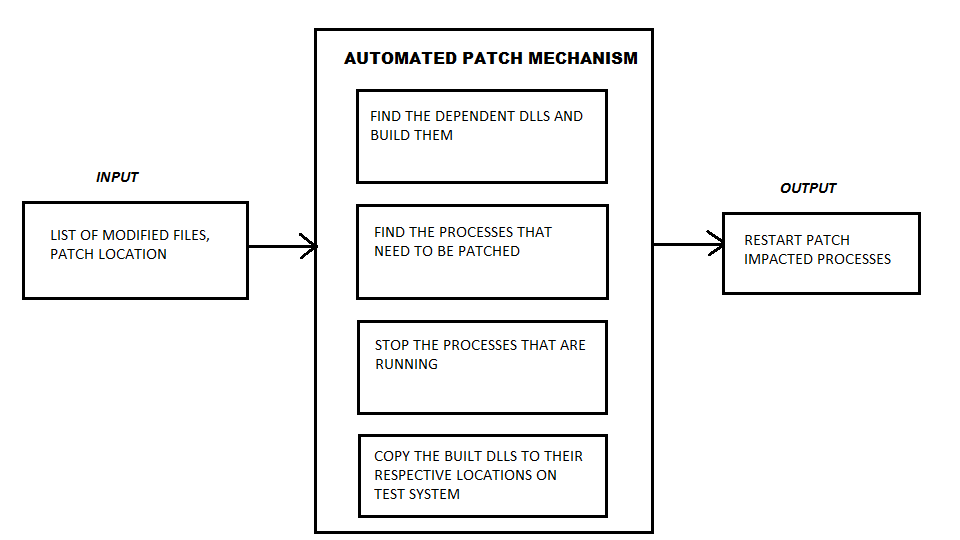
The patched processes are restarted on the remote test system.

* 1. Algorithm design

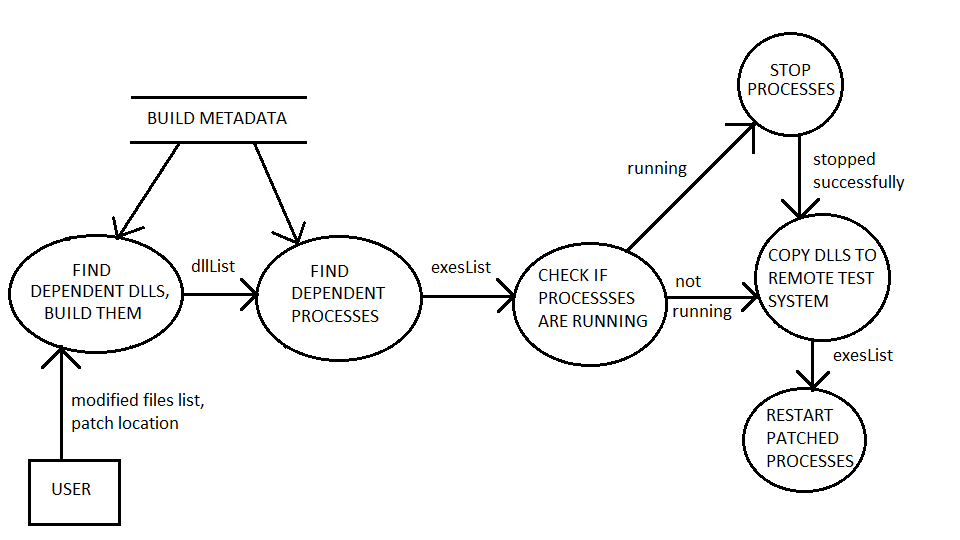
1. Input 🡨List of modified files and patch location
2. Use the build metadata to find the dlls that use the modified files(dllList)
3. For each dll**ϵ**dllList
4. Do
   * 1. Build the dll
     2. Using the metadata find the processes that need to be patched
     3. If the processes are running currently

Stop the processes

1. Copy the built dlls to the patch location on the remote test system
2. Restart patched processes
3. **Architecture Design**

****

1. **Data flow diagram**

****

1. **References**
2. Pankaj Jalote: A Concise Introduction to Software Engineering , Springer, 2008
3. Roger S. Pressman: Software Engineering A Practitioner's Approach, 7th Edition, McGraw Hill, 2010
4. David Gustafson: Software Engineering, Schaum's Outline Series, McGraw Hill, 2002
5. <https://en.wikipedia.org/wiki/Data_flow_diagram>
6. <https://www.edrawsoft.com/Data-Flow-Diagram-Symbols.php>
7. Gliffy diagrams