Abstract

This thesis considers Software Plug and Play as a potentially useful architectural abstraction that can solve many software architecture problems faced by vertical markets in the industry. While Plug and Play as a technology has been utilized to address many extensibility concerns in Information Technology (IT) infrastructure, there is no systematic support for similar extensibility notion at the software architecture level. To address these concerns, this thesis proposes the creation of frameworks and component models to support Plug and Play. Accordingly, this thesis considers Plug and Play as a user-defined framework, supported by a run-time component library that is embedded in the component schema, in order to support functional and structural extensibility of a running software system.

As practitioners, it was decided that the inductive path of Clayton Christensen’s approach for theory building [1-2] would be an appropriate way to build the architectural foundations for Software Plug and Play. Accordingly, this approach was adopted to empirically study six graded systems (discussed in chapters 5 to 10) that had decreasing requirements on their environments. Through this empirical and experimental work, the notion of software components, components composition and software architecture was illustrated. Further, the architecture principles that were adopted to enable plug and play in these systems aided in identifying the corresponding Plug and Play architectural abstractions. Based on this experimental experience, this thesis contributes the following to the field of computer science in general and software architecture in particular:

1. The concept that Software Plug and Play is a collection of standards, encoded in the run-time components and embedded in the component schema, for supporting extensibility of a running software system.
2. The characterization of software Plug and Play in terms of the underlying architecture principles, and corresponding architectural abstractions.
3. The formulation of a generic Plug and Play component model, that supports this characterization, which when instantiated aids in realizing pure Plug and Play architectures.
4. An expression of the Plug and Socket interfaces, that are constituent elements of the Plug and Play component model, by means of abstract data types.