ABSTRACT

This thesis discusses the design and development of knowledge based approach for automatic software document generation using design patterns. Natural language generation is a highly complex task whose automation is performed by accessing huge amount of knowledge sources such as domain knowledge, world knowledge and common sense knowledge. Software documentation is chosen as application for our work because of its closed ended domain i.e. even if the domain is not clearly specified the inputs required have to be properly identified for producing corresponding document. As in any other NLG system, the first task is content determination phase. Inputs such as concepts, entities, relations, structure, sub process, process, states, transitions, events, triggers, time, operations, source, destination, results, and roles are identified. Generally, inputs needed for documenting a project, are obtained from various software requirement specification techniques such as data flow diagrams, entity relationships diagrams, state transition diagrams and control flow diagrams. In the next module, identified inputs for generating a natural language text are organized using knowledge representation technique, frames. In this work, the frame representation scheme has been enhanced with specially designed new component called perspective descriptors, which tries to improve the epistemological status of the frame structure and adapts to changes in the software.

In order to represent the process sequentiality among the software entities, causal links have been introduced in the knowledge representation schema in addition to the hierarchical links that depict the structural organization among them. The next phase of software document generation is
planning. Adaptive planning is required to deal with producing documents at different levels. For automatic software document generation a generic standard which consists of Aim, Introduction, Purpose, Objective, Functional Behavior, Informational Behavior, Procedural details and all sub process details using document design patterns, has been used. The document design patterns are designed to satisfy the linguistic constraints also. Generally, patterns provide recurrent solutions. For each and every component of the software standard, specific patterns such as initiator, instantiator, illustrator, comparator, counter, fetcher etc. that matches with the linguistic constraints have been designed. The output of the planner is passed to the realization phase to obtain the complete and perfect document.

Realization, maps the identified content from the content determination phase organized using planner phase and generates actual text, which meets syntactic and semantic constraints. The realization tasks are: structure realization and linguistic realization. Structure realization deals with choosing mark-up to convey document structure and linguistic realization deals with insertion of function words, choosing correct inflection of content words, order words within a sentence and applying rules. Based on the three views viz. functional view, behavioural view and informational view, a generative grammar has been designed in order to obtain a proper and perfect software document.