1

Parliament: a module for parliamentary procedure software

BAYLE SHANKS AND DANA B. DAHLSTROM

28 April 2005

Abstract

Parliamentary procedure is a widely used system of rules for group decision making. We describe a reusable software module, *Parliament*, that implements the logic and bookkeeping of parliamentary procedure, given a precise specification of the rules. For use with the module, we have also created a partial working specification of *Robert's Rules of Order*. *Parliament* is designed to be embedded in applications, such as to support face-to-face meetings or to facilitate computer-mediated online deliberation.

1.1 Introduction

Parliament is an open-source software module written in Python[1] that can be used to build programs that follow or moderate the conduct of a deliberative assembly using parliamentary procedure. Parliament encapsulates logic and bookkeeping functions necessary for the function of parliamentary procedure, and can be embedded in applications for face-to-face meetings, or for synchronous or asynchronous computer-mediated communication.

Parliament's central functions track meeting state, such as pending motions, the relationships among them, and business already transacted. The outer application is responsible for informing the Parliament module about

[1] http://python.org/

book name.
editor names.

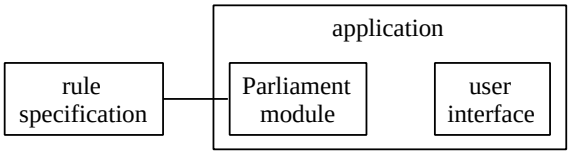

FIGURE 1 The Parliament module is embedded in an application, and uses an external rule specification.

events as they occur in the meeting, such as "member X made motion Y" or "motion Y failed." *Parliament* answers queries such as "Which motions are presently valid?" or "Which motions have been adopted in this meeting?" *Parliament* is also capable of answering questions about hypothetical situations, such as "Which motion will be pending if the immediately pending motion is adopted?"

The *Parliament* module does not incorporate the details of parliamentary procedure, such as the motions and customs described in *Robert's Rules of Order* (Robert (1915)). Instead, *Parliament* requires an external rule specification, allowing the user or developer to modify the rules independently and even to develop whole new rule systems.

An expanded version of this paper was presented at the Online Deliberation 2005 conference, and can be found at http://parliament.sf.net. The expanded version contains implementation details and reference material, including descriptions of data structures, the API, and the rule description language.

Using *Parliament*, we have also built a simple "Robert's-rules meeting assistant". For details about the meeting assistant, a survey of related work, and a discussion of design considerations for parliamentary software, please see our companion paper in this volume (Dahlstrom and Shanks (2006)).

1.2 A reusable module

Many software applications could share a common software implementation of parliamentary procedure; for example, applications that:

- (for online, synchronous meetings)
  - assist meeting participants during the meeting
  - assist meeting officers during the meeting
  - train people before the meeting[2]
  - provide a networked application which participants use to request the

[2]Non-modular parliamentary procedure training software called *Robert's Rules in Motion* is available at http://imovethat.com/

floor and to make and vote on motions

- (for online, asynchronous meetings —WWW or other)
  - assist a human chair
  - automatically chair a meeting
  - moderate a large discussion board or wiki according to formal meeting rules
  - automatically update a set of organizational bylaws according to the instructions of an online deliberative assembly

It would be inefficient to reimplement the core logic of parliamentary rules and sets of motions for each software project. Instead we offer the *Parliament* module for use in such applications.

1.3 Modular rule specifications

Robert's Rules of Order Newly Revised is the most common parliamentary authority in the United States, but there are others: older, public domain versions of *Robert's Rules of Order* and Sturgis's *Standard Code of Parliamentary Procedure* are examples. Each branch of the U.S. Congress uses its own rules, which are broadly similar to others just mentioned.

Rather than choosing a specific set of parliamentary rules and "hard-coding" them, we designed a single module to accommodate many different parliamentary rulesets. The user or developer may describe new rulesets in a specification language understood by the Parliament module, and the specification is then loaded at runtime (see Fig. 1). There are many advantages to this approach:

1. Different deliberative assemblies use different meeting rules.
2. Unconventional meeting settings such as the WWW or IRC demand new innovations in parliamentary rules.
3. Allowing the ruleset to be modified gives the assembly complete flexibility to adapt the software to their needs. Assemblies should not be forced to follow a particular set of meeting rules just because their software can't support the rules that they would really prefer.
4. Research on group decision-making support systems (GDSS) is hindered by the difficulty of isolating the effect of individual components of the group decision-making process. A configurable parliamentary ruleset will serve as the ideal platform for testing fine-grained modifications to a group's process.

1.4 The rule specification language

The rule specification language has a quasi-English syntax. A ruleset is typically written in a separate file and then loaded into the *Parliament* module

upon initialization.

The ruleset is specified in terms of “actions” that participants can take at certain times in the meeting.

The ruleset specification can be arbitrarily expressive; if there is no other way to express some desired behavior, arbitrary Python code can be embedded into the ruleset.

Examples of the rule specification language

Here’s how the motion to “Lay on the table” is defined in the Robert’s-rules ruleset (note that the actual definition has a longer summary and is not word-wrapped):

```
-----------------
NAME: Lay on the table
MOTION TO FORM OF NAME: "Motion to lay
on the table"

TYPE: Subsidiary motion

SUMMARY: "The objective of this motion is to
temporarily lay a question aside"

motion precedence: 1
debatable: NO
amendable: NO
subsidiaries allowed: NO
reconsiderable: ONLY WHEN (WAS_ACCEPTED)

TARGET: ancestor motion
ON PASS: table target

category: "scheduling"
purpose: "delay"

# comments can be embedded like this

RRO section ref: "19"
RROR section ref: "28"

{
def example_method(self):
    print ’This is embedded Python code’
```

```
}

-----------------
```

1.5 The Robert's-rules parliamentary ruleset specification

We have written a partial ruleset specification for the 1915 (now public-domain) Fourth Edition of *Robert's Rules of Order Revised*. The ruleset includes over 25 of the most common motions and their important attributes—such as when they are debatable and what vote is required for them to carry—as well as most of the precedence relations between the motions and some of their semantics.

This specification was initially based on Henry Prakken's formalization of the Rules (Prakken (1998)), which he kindly provided to us in machine-readable form. We made many changes to Prakken's formalization, including the addition of the complicated logic of precedence.

Lessons learned during the formal specification of Robert's rules may be found in the expanded version of this paper.

1.6 Conclusion

The *Parliament* module provides the central infrastructure for parliamentary-procedure software. A reusable module, it implements and interprets parliamentary rules, tracks meeting state, and infers important information such as which motions are in order at a given time. The module is flexible, accomodating any properly codified set of parliamentary rules.

A concise specification language allows others to create and modify rulesets efficiently. A usable partial specification of Robert's rules has been created, and a prototype Robert's-rules meeting assistant has been built and used in real face-to-face meetings (Dahlstrom and Shanks (2006)).

The *Parliament* module shows potential for use in many contexts, including both face-to-face and online meetings. It is hoped the module will lead to a variety of useful parliamentary software.

References

Dahlstrom and Shanks. 2006. *Software support for face-to-face parliamentary procedure*. (in this volume).

Prakken, Henry. 1998. Formalizing robert's rules of order: An experiment in automating mediation of group decision making. Tech. Rep. REP-FIT-1998-12, GMD.

Robert, Henry M. 1915. *Robert's Rules of Order Revised*. Public Domain.