

ROTORCRAFT MODEL DEVELOPMENT UTILIZING REVISION CONTROL SYSTEM

Jinwei Shen
National Institute of Aerospace
Hampton, VA

EXTENDED ABSTRACT FOR AHS 65TH ANNUAL FORUM - SYSTEM ENGINEERING SESSION

Objective

Rotorcraft consist of complex mechanical and structural systems that include interconnected rigid and deformable components. The dynamics of such large-scale, multibody systems, which are further loaded with unsteady aerodynamics forces, is highly nonlinear, non-periodic, and difficult to predict (Ref. 1). Multibody Dynamics Simulation (MDS) methodology, has been adopted for rotorcraft applications in recent years (Ref. 2,3). Coupled with Computational Fluid Dynamics (CFD) analyses, which provide accurate predictions of the aerodynamic forces, Computational Structural Dynamics (CSD) models of rotorcraft based on MDS establish a high fidelity simulation framework for studying rotorcraft aeromechanics (Ref. 4). However, the complexities of such sophisticated rotorcraft models increase dramatically, and the development of such a model will progress in stages, and with iterations. Such a modeling development procedure becomes more similar with the code development project in software industry. Thus, the standard practice, Reversion Control Systems (RCS), used in source code management to assure the integrity, maintainability, and extendability of its projects, can also be applied to managing the rotorcraft model development.

This paper presents a software engineering tool: a state-of-the-art revision control system, and demonstrates its application in rotorcraft model development, especially on its facilitation of the modular model development approach. The case studies include in this paper are a validation study: WRATS semi-span tiltrotor whirl flutter investigation, and two conceptual design study of a UAV tiltrotor and a mono-tiltrotor. DYMORE (Ref. 5), a multibody-based comprehensive rotorcraft analysis, is used to develop the rotorcraft models. Git (Ref. 6), an open source version control system designed to handle very large projects with speed and efficiency, is used to manage the model development procedure. The benefits of using revision control in sophistic rotorcraft model development will be demonstrated in various common scenarios arising in such projects.

Approach

Revision control is the management of multiple revisions of the same unit of information. It is routinely used in managing ongoing development of source code and other digital documentation in software industry. Various revision control software have been used for years. They provided the basic capabilities of a revision control system, however, they are rather primitive in their design, and require careful maintenance itself. State-of-the art RCS, such as Git, improves on the earlier RCS implementations, provides a comprehensive revision control system, minimizes distractions from the project development, requires less maintenance, and provides fast operation and better usability.

Basic capabilities of a revision control system

The basic capabilities of a revision control system are “reversibility”, “concurrency”, and “history”. Reversibility is the ability to revert to an earlier known-good state. Concurrency is the ability to have many people modifying the same collection of documents knowing that conflicting modifications can be detected and resolved. History is the capability to attach a history to your data, explanatory comments about the intention behind each change to it.

Improvements made by Git

Git is a state-of-the-art revision control system designed by Linus Torvalds, the author of Linux kernel, and for the management of the Linux kernel source code, a large project. Its development began in 2005, and achieved maturity in 2007. Git has many advantages over the earlier generations of revision control systems as well as its contemporaries.

Git has unique features in the three basic capabilities of a RCS. In “reversibility”, Git has a build-in algorithm that guarantees that the model files checked-out are identical to those initially checked-in. In “concurrency”, Git adopts a distributed development approach. It provides each developer a local copy, and thus local changes are unaffected by other developer’s modifications, but in the mean time, makes the merging of those local changes fast and easy when ready. In “history”, Git tracks the project content instead of individual files, and this enables it tracks the modification history in fine details. Git has advantages over some contemporary revision control systems. Most RCS requires the developer to be constantly mindful of the state of current project in the revision control system when developing models and making revisions. Git, however, provides capabilities to separate model development and version control, and let the developer focusing on model development and worrying about revision control late. It provides mechanism to solve the tangled working copy problem. Furthermore, Git is well-known for its performance. A fast revision control system will change the developer’s work-flow and increase his productivity.

Managing rotorcraft model development with Git

The information contents in a sophistic rotorcraft dynamics model is as intricate and inter-connecting as a software package, and as such, the advantages of managing it with a revision control system are apparent. Some of these benefits are listed below:

1. promoting the modular model development approach. Model modules can be developed and validated separately, and merged back into the main model and vise versa.
2. fostering group model development by allowing revision isolation and merging.
3. expediting the model development iterations by eliminating the unnecessary revision duplication.
4. increasing “portability”. RCS can push the revisions of a model to other similar model, i.e. from a scale-demonstrator to a full-scale model.
5. maintaining “repeatability”. RCS can keep the model files with its running scripts, data retrieving, post-processing, and visualizing scripts in sync. When reverting model to an earlier version in history, this ensures the simulation results repeatable.

Case Study

Several common scenarios in rotorcraft model development will be used to demonstrate the power of revision control system. The rotorcraft models shown in these case studies are developed with DYMORE. DYMORE is a finite element based tool for the analysis of nonlinear flexible multibody systems, developed at the School of Aerospace Engineering, Georgia Institute of Technology. It is capable of supporting comprehensive, multibody-based dynamic analyses of rotorcraft modeling and simulation.

WRATS stiff-inplane tiltrotor validation

—scenario: implementing modular model development approach

The author presented in detail a validation study of WRATS (Wing and Rotor Aeroelastic Testing System) tiltrotor semi-span model in Ref 3. The study adopted a modular approach which is facilitated by MDS, and enables a systematic validation process of the analytical model. Modular procedure is another pragmatic approach borrowed from software industry. Multibody dynamics analyses provide elementary structural components such as rigid body, beam, cable, and shell elements as well as constraint components including the revolute hinge, prismatic joint, spherical joint, and boundary condition constraint. These features facilitate the development of sophisticated structural rotorcraft models component by component, as well as the validation of the model component by component.

Fig. 1 illustrates the modular procedure applied to the semi-span WRATS stiff-inplane tiltrotor model. The model can be naturally divided into two sub-systems, namely, the fixed sub-system and rotating sub-system which are developed and validated separately. Ref 3 demonstrated that modularization in model development and validation is a reliable method to build the analytical model accurately and efficiently. However, this modular approach will entail duplicated work if adopted without using a revision control system. For example, a tiltrotor hub model may be developed on an isolated rotor configuration through iterations. These model revisions will have to be duplicated on the semi-span model manually to study the effects of the improved design with every iteration. However, with a revision control system, these changes can be easily merged between the two configurations.

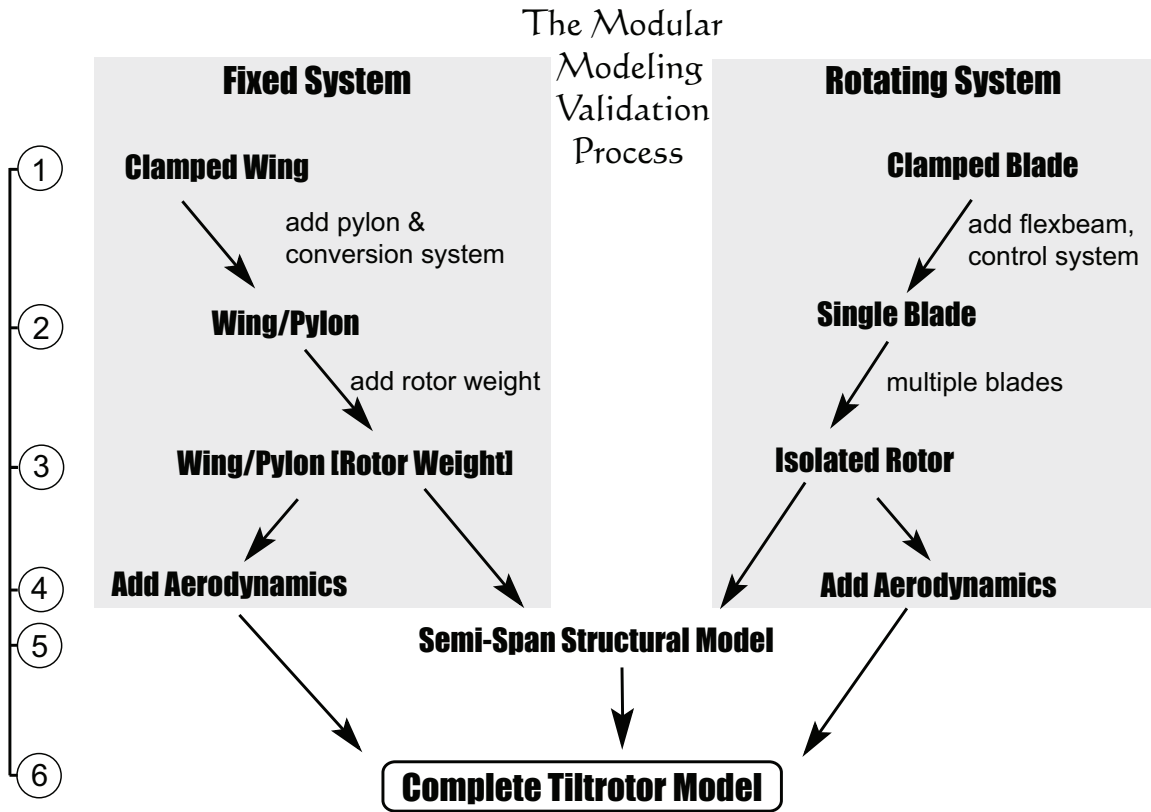


Figure 1: The modular modeling and validation process.

UAV tiltrotor design study

—scenario: developing a semi-span and a full-span model

Ref 7 presents a load and stability study for a proposed unmanned tiltrotor aircraft in support of the conceptual design process. The stability calculations are obtained for both the semi-span and full-span model (Fig. 2.) The semi-span model is developed first, and then extended to the full-span model. There are a lot of similarities between the right and left semi-span components. Nonetheless, they are not identical, instead, mirror to each other. A technology available in Git, called “cherry-picking”, is used to push the changes made to the right hand semi-span model to the left hand semi-span model.

Mono-tiltrotor design study

—scenario: developing model in different scales

Modeling development is carried out for both the scale-demonstrator and the full scale mono-tiltrotor, and also involves different flight configurations (Fig. 3.) Git will be used to assist the transferring of modifications among the the two scale models and among branches of the same model (different configurations) through the design iterations.

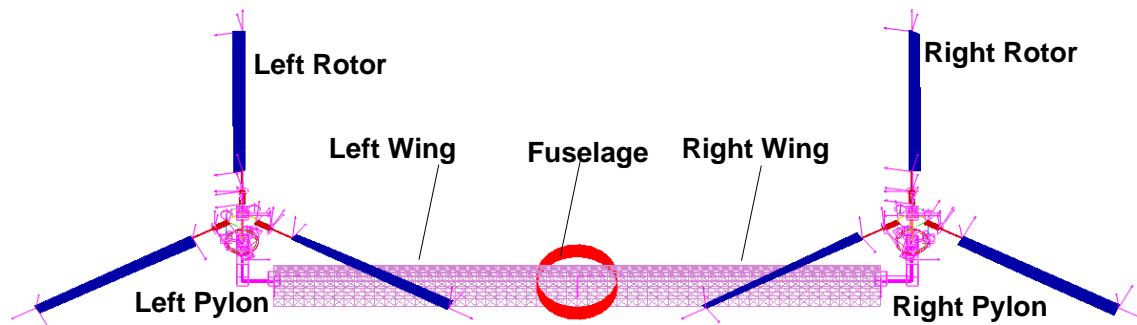


Figure 2: Full-span UAV tiltrotor dynamics model.

Full Paper

In the full paper, a state-of-the-art revision control system will be utilized in rotorcraft model development. The benefits of using revision control in managing model development will be demonstrated with various common scenarios arises in such projects. The paper will especially emphasize how the revision control system enhances the modular approach in sophistic rotorcraft model development.

References

- ¹Yeo, H. and Johnson, W., "Assessment of Comprehensive Analysis Calculation of Structural Loads on Rotors," Proceedings of AHS 60th Annual Forum, Baltimore, MD, June 7-10 2004, p. 26.
- ²Hodges, D. H., Saberi, H., and Ormiston, R. A., "Development of Nonlinear Beam Elements for Rotorcraft Comprehensive Analyses," *Journal of the American Helicopter Society*, Vol. 52, (1), Jan. 2007, pp. 36–48.
- ³Shen, J., Masarati, P., Roget, B., Piatak, D. J., Nixon, M. W., and Singleton, J. D., "Modeling A Stiff-Inplane Tiltrotor Using Two Multibody Analyses: A Validation Study," Proceedings of AHS 64th Annual Forum, Montreal, Canada, April 29 - May 1 2008, p. 9.
- ⁴Bhagwat, M. J., Ormiston, R. A., Saberi, H. A., and Xin, H., "Application of CFD/CSD Coupling for Analysis of Rotorcraft Airloads and Blade Loads in Maneuvering Flight," Proceedings of AHS 63rd Annual Forum, Virginia Beach, VA, May 1-3 2007, p. 30.
- ⁵Bauchau, O., Bottasso, C., and Nikishkov, Y., "Modeling Rotorcraft Dynamics with Finite Element Multibody Procedures," *Mathematical and Computer Modeling*, Vol. 33, 2001, pp. 1113–1137.
- ⁶Torvalds, L., *Git User's Manual*, first edition, September 2008.
- ⁷Floros, M., Shen, J., Lee, M. K., , and Hwang, S., "Loads and Stability Analysis of an Unmanned Tilt Rotor," American Helicopter Society 62nd Annual Forum Proceedings, Phoenix, AZ, May 2006, p. 19.

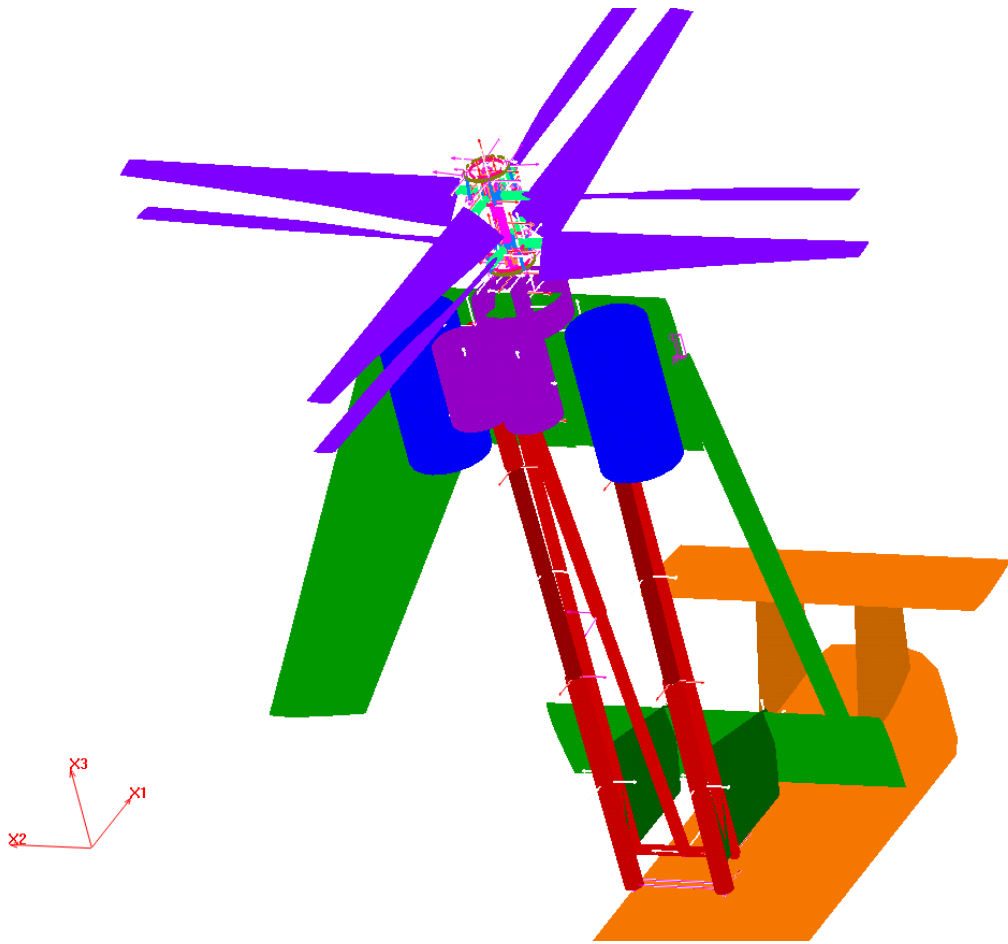


Figure 3: Mono-tiltrotor dynamics model.