California Institute of Technology

Software Reliability Measurement

Allen P. Nikora
Jet Propulsion Laboratory,
California Institute of Technology
Pasadena, CA 91109
Allen.P.Nikora@jpl.nasa.gov

Myron Hecht
The Aerospace Corporation
2350 E. El Segundo Bl.
El Segundo, CA 90245
Myron.J.Hecht@aero.org

Douglas J. Buettner
The Aerospace Corporation
2350 E. El Segundo Bl.
El Segundo, CA 90245
Douglas.J.Buettner@aero.org

Agenda

- What is Software Reliability and Why Do We Care?
- Measuring and Estimating Software Reliability
  - “Classical” Software Reliability Modeling
  - Fault Modeling Prior To Test
- Where Do We Go From Here?

What is Software Reliability?

California Institute of Technology

Software reliability: “The probability that software will not cause the failure of a system for a specified time under specified conditions. The probability is a function of the inputs to and use of the system as well as a function of the existence of faults in the software. The inputs to the system determine whether existing faults, if any, are encountered”. [IEEE89]

THE AEROSPACE CORPORATION

Why Do We Care?

- Determine whether software can be released
- Predict resources required to bring software to required reliability
- Determine impact of insufficient resources on operation reliability
- Prioritize testing/inspection of modules having highest estimated fault content
- Develop fault-avoidance techniques:
  - Minimize number of faults inserted
  - Prevent insertion of specific types of faults

Measuring and Estimating Software Reliability

California Institute of Technology

- "Classical" software reliability modeling
  - Statistical models applied during software test can estimate/forecast reliability
    - Inputs:
      - Time between successive failures, or
      - Number of failures per test interval of a given length
    - Outputs:
      - Probability Density Function (PDF) of time to next failure, or
      - PDF of number of failures in next interval
      - Can estimate/forecast reliability, failure intensity, number of failures observed in next **n** intervals, …

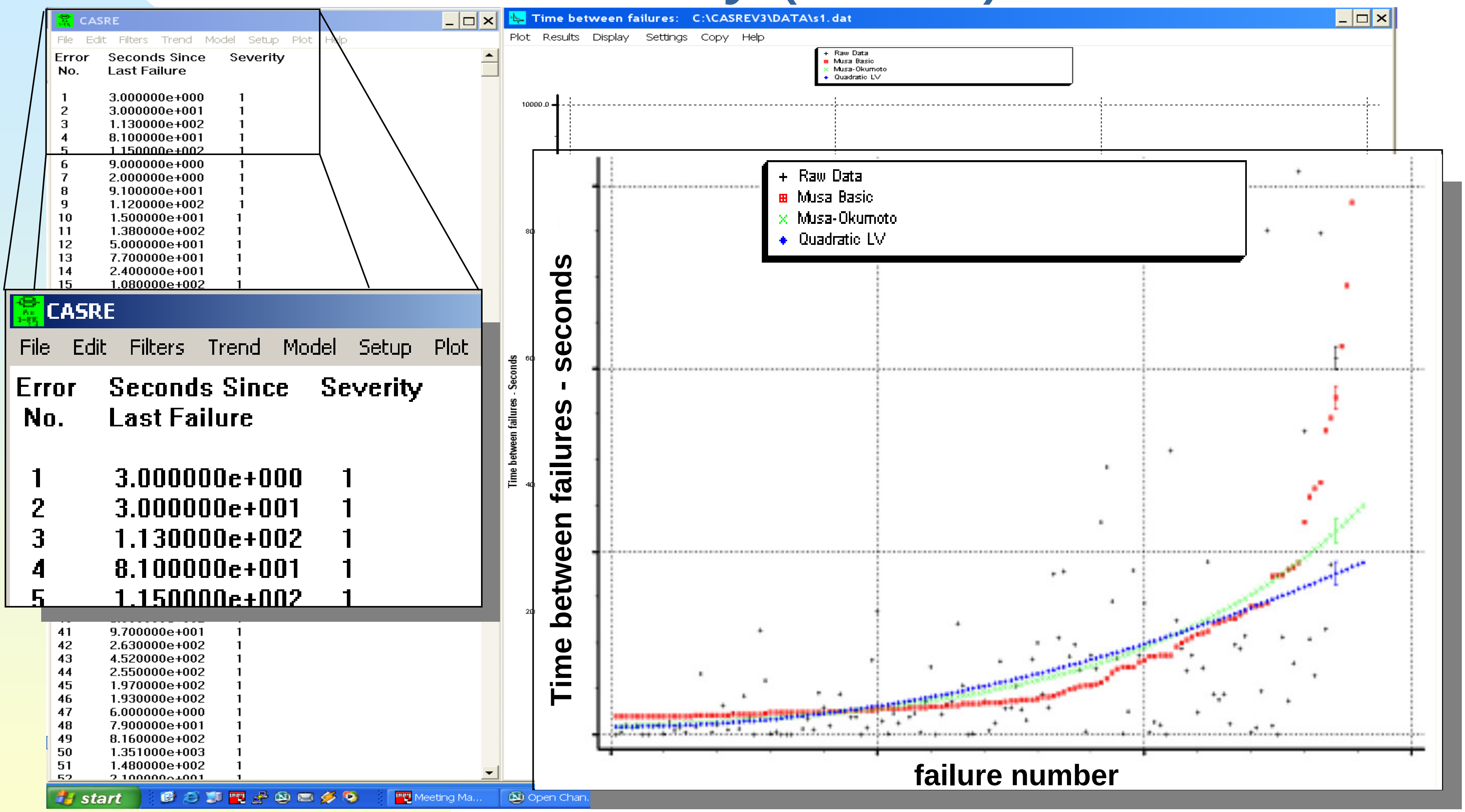
THE AEROSPACE CORPORATION

Measuring and Estimating Software Reliability (cont'd)

- "Classical" software reliability modeling
  - Over 100 models published since 1971. Better known models include:
    - Musa Basic
    - Musa-Okumoto
    - Schneidewind (used on Space Shuttle software)
    - Littlewood-Verrall
    - Goal-Okumoto (NHPP for interval failure counts)
  - Following slide shows sample input/output for time between successive failures model.
    - Modeling done using CASRE version 3.0. CASRE is available free of charge from the Open Channel Foundation, "http://www.openchannelfoundation.org/projects/CASRE_3.0"

Measuring and Estimating Software Reliability (cont'd)

California Institute of Technology



Example of CASRE software reliability models

THE AEROSPACE CORPORATION

Measuring and Estimating Software Reliability Prior to Test

- Identifying Fault-Prone Modules
  - Boolean Discriminant Functions [Schn97]
  - Classification Trees
    - Khoshgoftaar and Allen [Khos01a]
    - Ghokale and Lyu [Gokh97]
  - Logistic Regression [Schn01]

Measuring and Estimating Software Reliability Prior to Test (cont'd)

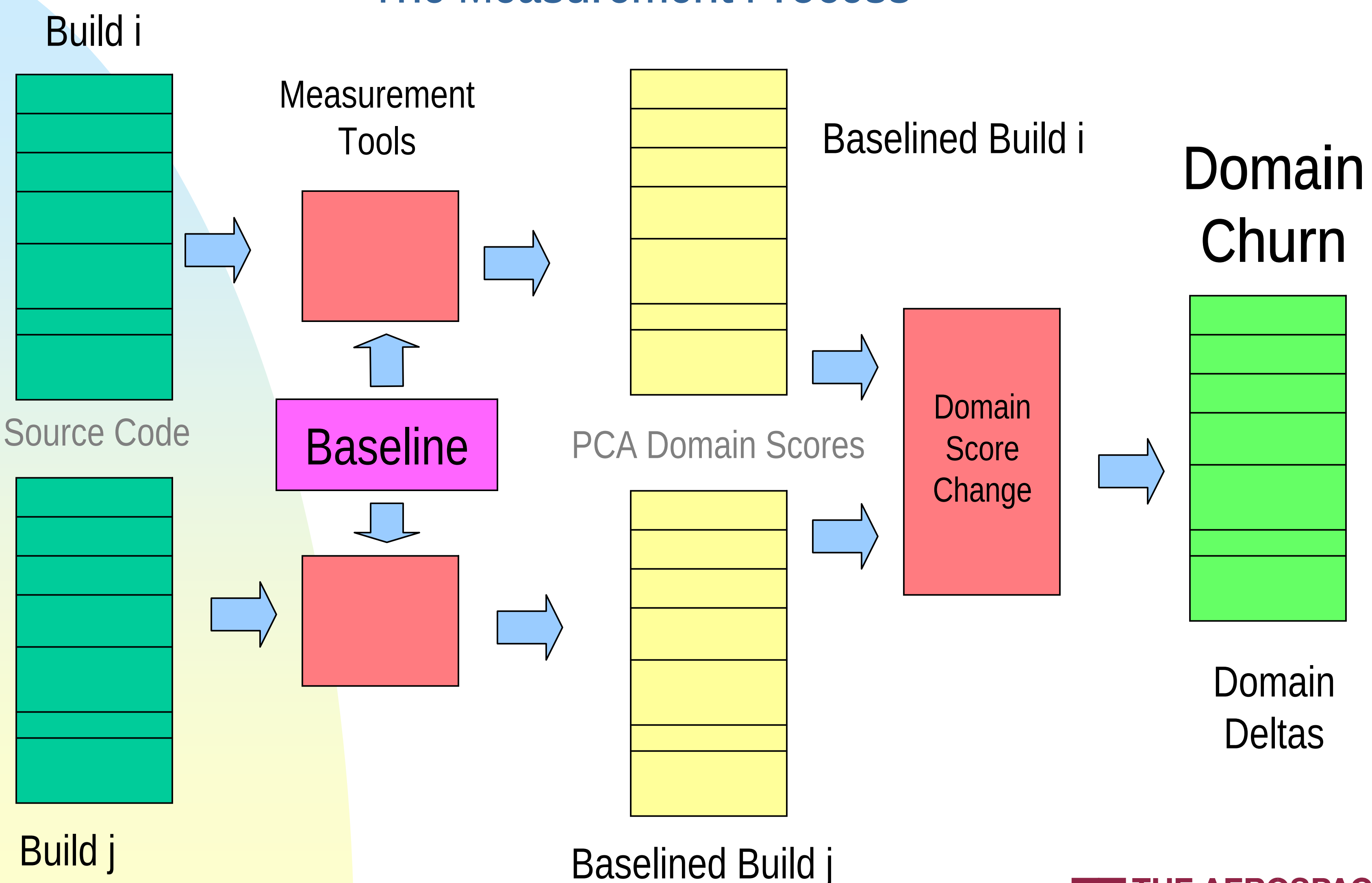
California Institute of Technology

- Estimating software fault content – representative efforts include:
  - Module-order modeling
  - Neural nets
  - Zero-inflated Poisson regression [Khos01]
  - Measurement of structural evolution [Niko03], [Niko98]
  - …

THE AEROSPACE CORPORATION

Measurement of Structural Evolution

The Measurement Process



PCA = Principle Component Analysis

Measurement of Structural Evolution

View of Structural Evolution at the System Level

Graph of Code Churn and Code Delta in th...

Darwin
Software Systems International

Darwin Portal

Navigation

Darwin Main

Manager Information

Tester Information

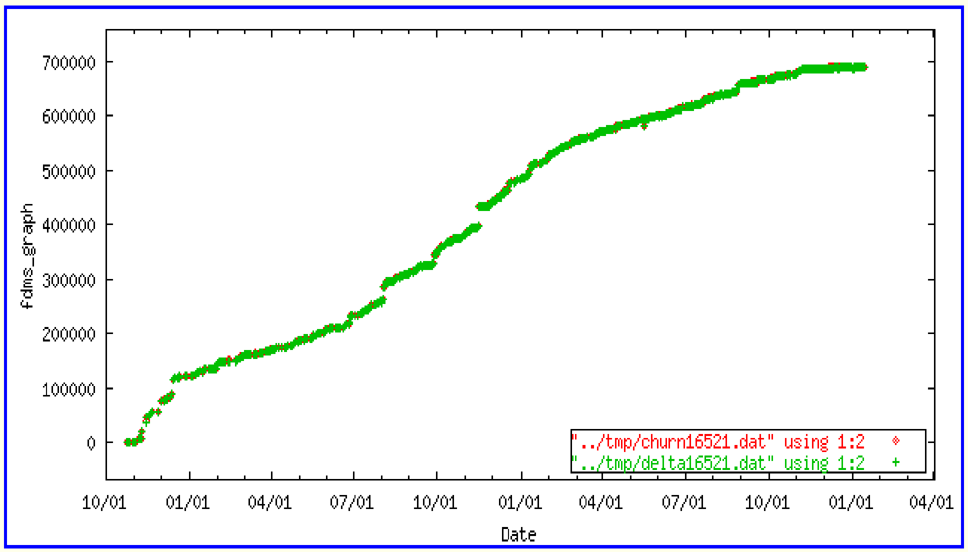
Education

Project Manager

View Available Databases

Feedback

Graph of Code Churn and Code Delta for the project fdms_project_1b.



Click here for help

documents/Metrics2003cfp.pdf

California Institute of Technology

THE AEROSPACE CORPORATION

Measurement of Structural Evolution

View of Structural Evolution at the Module Level

(Non-zero) Modules for build 2001-11-02 o...

(Non-zero) Modules for build 2001-11-02 of project project_1b, sorted by Churn since baseline.

Modulename	Churn From Baseline
doContent(XML_Parser parser, int startTagLevel, const ENCODING *enc, const char *s, const char *end, const char **nextPtr)	329.523165
doProlog(XML_Parser parser, const ENCODING *enc, const char *s, const char *end, int tok, const char *next, const char **nextPtr)	311.585820
examples()	309.099387
processMeasurementAndPredict(const Mds::Fw::Time::Tmgt::RTEpoch& current, const Mds::Fw::Time::Tmgt::RTEpoch& stop)	289.353008
test2s()	279.141671
TestDiscrete::TestDiscrete()	260.394345
TestIntervallic::TestIntervallic()	256.865234
storeAtts(XML_Parser parser, const ENCODING *enc, const char *attStr, TAG_NAME *tagNamePtr, BINDING **bindingsPtr)	240.951958
GreaseFilterTest(Dispatch& r, const std::string& key, const CGIArgs& args)	240.699311
doTest()	237.752957
PositionEstimateFunctionTest(Dispatch& r, const std::string& key, const CGIArgs& args)	223.900440
DirectedGraph::close()	214.416
SimpleNormalPositionEstimatorTraits::Thread::updateStateVariables(const Mds::Fw::Time::Tmgt::RTEpoch& start, const Mds::Fw::Time::Tmgt::RTEpoch& stop, const Mds::Fw::Filter::Grease::GreaseBasis& state, const Mds::Fw::Filter::Grease::GreaseBasis& covariance, const Mds::Rd::Mars::Common::SimpleAirDragModel::AirDragModelParameterType& air_drag_model_para, double spacecraft_mass, double avg_engine_thrust)	204.313
AirDragModelParameterEstimatorTraits::Thread::predictState()	203.781
ParachuteEstimatorTraits::Thread::predictState()	195.118
SimpleNormalPositionEstimatorTraits::Thread::changed(const Mds::Fw::Cmp::RefCountComponentInstance monitored_sv, Mds::Fw::Dm::Vhis::ConstItemVectorRef changedItems)	182.312
PREFIX(prologTok)	175.659645
LengthTest(Dispatch& r, const std::string& key, const CGIArgs& args)	165.230353
vxmain(int ctor, const char* argList)	164.506422
GoalNetTestHarness::createXGoalNet(Dispatch& r, const std::string& key, const CGIArgs& args)	163.479793
vxmain_internal(const char* argList)	162.467209

Transferring data from swmeasurement.jpl.nasa.gov...

Ranked ordering by module of code churn per software build

JPL California Institute of Technology

THE AEROSPACE CORPORATION

Measuring and Estimating Software Reliability Prior to Test (cont'd)

- Architecture-based reliability estimates
  - Architecture-based Reliability Risk Analysis [Ammar2002]
  - Comparison of Architecture-based Software Reliability Models [Gos2001]
  - Cost vs. Reliability Architectural Tradeoffs [Gokh99]

California Institute of Technology

THE AEROSPACE CORPORATION

Predicting Software Reliability Prior to Test (cont'd)

California Institute of Technology

- Architecture-based reliability prediction [Li97]
  - Model software system as a communicating extended finite state machine
  - Map the software architecture into a high-abstraction level design language such as Petri-nets or the Specification and Description Language (SDL)
  - Simulate elements in the system such as work-flow activation, failure occurrence, throughput, and of course the behavior and flows of the architecture from the architectural model
  - Use simulator to analyze the reliability of architectural designs
  - Experiments performed on a Bellcore product
  - Experimental results demonstrated the ability to predict the reliability of a software architecture
  - Verification of the predicted reliability not performed

THE AEROSPACE CORPORATION

Where Do We Go From Here?

California Institute of Technology

- Current techniques allow:
  - Estimation/forecasting of software reliability during test and operations
  - Identification of faulty modules during implementation, prior to test
  - Fault content estimates during implementation, prior to test
  - Evaluation of architectures with respect to reliability

THE AEROSPACE CORPORATION

Where Do We Go From Here?

California Institute of Technology

- Future work
  - Effect of design/requirements changes to reliability
  - Effects of software characteristics, development process characteristics on quality
    - Constructive Quality Model COQUALMO under development at USC Center for Software Engineering – see "http://sunset.usc.edu/research/coqualmo/index.html"
  - Verification of predicted architectural reliability from simulations
    - Aerospace IR&D, "Space Systems Mission Assurance via Software Reliability Monitoring"
  - …

THE AEROSPACE CORPORATION

References and Further Reading

California Institute of Technology

[Ammar02] Sherif M. Yacoub, Hany H. Ammar , “A Methodology for Architecture-Level Reliability Risk Analysis,” IEEE Transactions on Software Engineering, June 2002, pp. 529-547

[Gokh99] S. Wadekar and S. Gokhale, "Exploring Cost and Reliability Tradeoffs in Architectual Alternatives Using a Genetic Algorithm," in Proc. of Intl. Symposium on Software Reliability Engineering (ISSRE 99), pp. 104-113, Boca Raton, FL, November 1999.

[Gokh97] S. S. Gokhale, M. R. Lyu, “Regression Tree Modeling for the Prediction of Software Quality”, proceedings of the Third ISSAT International Conference on Reliability and Quality in Design, pp 31-36, Anaheim, CA, March 12-14, 1997

[Gos2001] K.Goseva-Popstojanova, A.P.Mathur, K.S.Trivedi, Comparison of Architecture-Based Software Reliability Models, Proc. 12th IEEE International Symposium on Software Reliability Engineering (ISSRE 2001), Hong Kong, Nov. 2001.

[IEEE89] IEEE Guide for the Use of IEEE Standard Dictionary of Measures to Produce Reliable Software, IEEE Std 982.1-1988, Institute of Electrical and Electronics Engineers, 1988.

[Khosh01] T. Khoshgoftaar, "An Application of Zero-Inflated Poisson Regression for Software Fault Prediction", proceedings of the 12th International Symposium on Software Reliability Engineering, pp 66-73, Hong Kong, Nov, 2001.

THE AEROSPACE CORPORATION

References and Further Reading

California Institute of Technology

[Khosh01a] T. M. Khoshgoftaar, E. B. Allen, "Modeling Software Quality with Classification Trees", in H. Pham (ed), Recent Advances in Reliability and Quality Engineering, Chapter 15, pp 247-270, World Scientific Publishing, Singapore, 2001.

[Li97] J. Jenny Li, Josephine Micallef, and Joseph R. Horgan, "Automatic Simulation to Predict Software Architecture Reliability", Proc. 8th IEEE International Symposium on Software Reliability Engineering (ISSRE 1997), Albuquerque, NM, Nov. 1997.

[Niko2003] A. Nikora, J. Munson, "Developing Fault Predictors for Evolving Software Systems", proceedings of the 9th International Software Metrics Symposium, Sep 3-5, Sydney, Australia

[Niko98] A. P. Nikora, J. C. Munson, "Determining Fault Insertion Rates For Evolving Software Systems", proceedings of the 1998 IEEE International Symposium of Software Reliability Engineering, Paderborn, Germany, November 1998, IEEE Press.

[Schn01] N. F. Schneidewind, "Investigation of Logistic Regression as a Discriminant of Software Quality", proceedings of the 7th International Software Metrics Symposium, pp 328-337, London, April, 2001.

[Schn97] N. F. Schneidewind, "Software Metrics Model for Integrating Quality Control and Prediction", proceedings of the 8th International Symposium on Software Reliability Engineering, pp 402-415, Albuquerque, NM, Nov, 1997.

THE AEROSPACE CORPORATION