Proceedings of the 2005 Winter Simulation Conference
M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds.

DATA FARMING: DISCOVERING SURPRISE

Gary E. Horne

Referentia Systems Incorporated
550 Paiea St.
Honolulu, HI 96819, U.S.A.

Theodore E. Meyer

The MITRE Corporation
1719 Linwood Place
McLean, VA 22101, U.S.A.

ABSTRACT

Data Farming is a methodology and capability that makes use of high performance computing to run models many times. This capability gives modelers and their clients the enhanced ability to discover trends and outlier in results, do sensitivity studies, verify and validate over extended ranges of input parameters, and consider modeling and analyzing non-linear phenomena with characteristics that can not be precisely defined. As high performance computing, in the form of distributed computing capabilities and commodity node systems becomes more pervasive and cost effective, Data Farming can become more available to modelers. In this paper the authors summarize Data Farming and the processes and data architecture of Data Farming systems that make high performance computing readily available to modelers.

1 INTRODUCTION TO DATA FARMING

Data Farming is a process that was developed to aid decision-makers in answering questions that are not addressed by traditional modeling and modeling processes. Data Farming has been used to seek insight into questions such as:

- What is the role of trust, or other so-called 'intangibles', on the battlefield?
- What impact will net-centric warfare and complete information sharing have on the effectiveness of military units?
- How can we best protect our homeland from a martyr-based offense?
- How can a bio-terrorist attack be mitigated in a free society?
- What system characteristics are important in military convoy protection systems?
- What factors are most important in the development of martyrs or terrorists?

Of course, there are many other questions which are of interest, and these are but a few of the ones that teams have attempted to address using Data Farming. (Horne 2001).

These types of question can never have precisely defined initial conditions and a complete set of algorithms that describe the system being considered. These questions address open systems that defy prediction. Data Farming is used to provide insight that can be used by decision-makers. To accomplish this Data Farming relies upon two basic ideas:

1. use high performance computing (HPC) to execute models many times over varied initial conditions to gain understanding of the possible outliers, trends, and distribution of results, and
2. develop models, called distillations, that are focused to specifically address the question.

Data Farming, by providing the ability to process large parameter spaces, makes possible the discovery of surprises (both positive and negative) and potential options.

Data Farming is an iterative team process (Horne and Meyer 2004). Figure 1 presents the data farming process as a set of imbedded loops. This process normally require input and participation by subject matter experts, modelers, analysts, and decision-makers.

The "Scenario Creation" loop shown on the left side of the figure involves developing and honing a model that adequately represents the system that addresses the question being asked by the decision-maker. This is an iterative process that often requires honing the question as well.

The "Scenario Run Space Execution" loop shown in Figure 1 is entered once the *basecase* of the scenario is complete. In this loop the team defines a *study* which determines which scenario input parameters should be examined and what processes should be used to vary them. Here the team is exploring the possible variations (or *excursions* of the basecase) in the initial conditions of the scenario. Specifically those parameters that address the question being posed are considered.

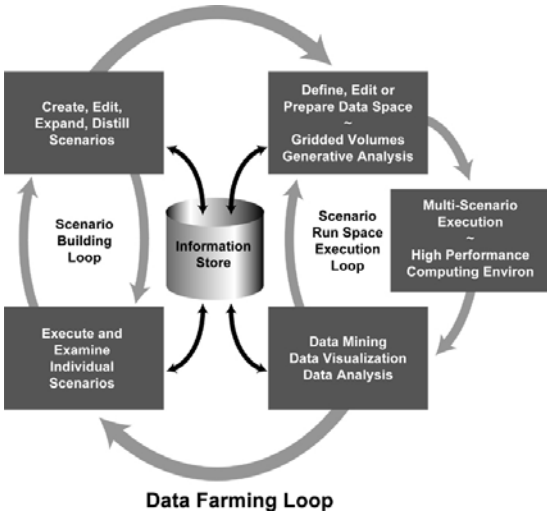

Figure 1: Data Farming Iterative Process

The defined study is used to guide the execution of many runs of the model in the HPC environment. Each run produces output which is collected by the data farming system and provided as output to analysis capabilities. On analysis of the results, the team (or an algorithm) may decide to adjust or produce a new study or adjust the model to more adequately address the question.

This process continues until insight related to the decision-maker's question has been gained.

Data Farming continues to evolve. Project Albert (Brandstein and Horne 1998), a United States Marine Corps project aimed at prototyping and experimenting with Data Farming methodologies has resulted in the development of a variety of components. These prototype components, including distillation modeling systems, HPC model execution systems, and results analysis systems, continue to evolve and be developed by various Project Albert collaborators. As this development has continued, though, the requirements for the data that passes between the components has become better understood. informal "standards" have emerged that allow developers of the components to evolve there capabilities.

2 DATA ARCHITECTURE FOR DATA FARMING?

The "standards" used by existing collaboratively developed Data Farming systems do not form a well defined Data Farming data architecture. The intent of this paper is to present an overview of the current state of the data types, structures, objects, and paths used in Data Farming to begin a dialogue aimed at producing a base set of evolvable, flexible and extensible data standards. Historically, the definition of a set of evolvable base data standards has lead to the expansion and popularizing of technical capabilities. Examples of this include HTML and the Web, Postscript and desktop publishing, SQL and relational databases, and ASCII and text processing. By defining an initial set of data standards the applications and components used for Data Farming (or any technical capability) can be improved, made easier to use, expanded, or replaced within a collaborative, competitive, and evolving environment.

Where Figure 1 is an illustration of the process of data farming, Figure 2 is a "strawman" diagram of the system components of a Data Farming system and the flow of data through that system.

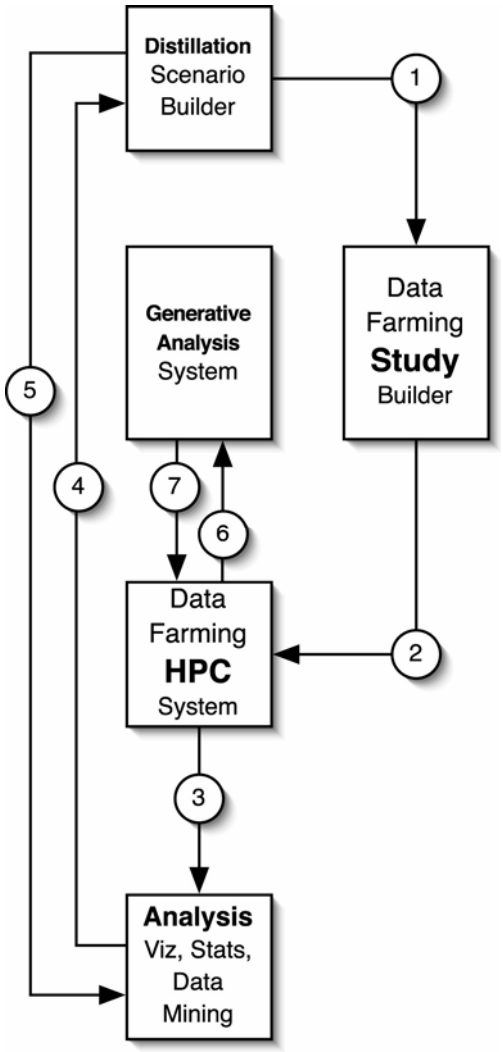

Figure 2: Data Farming Components and Data Paths; All Components Can Also Take Potential User Input

By examining and understanding the data interchanges in this system, it is hoped that a dialogue to define the basic data standards might begin.

2.1 XML and HDF

Two "formal" data standards have been adopted by current Data Farming collaborators: the eXtensible Markup Language (XML) and the Hierarchical Data Format (HDF). XML (W3C 2003) is well known and pervasively adopted by the information technology community as a method for exchange of metadata and attributed data structures. It will not be described here. HDF (HDF 2005) is less well known, but is pervasively used by science communities that store, examine and manipulate large, complex data structures.

HDF is a binary, cross-platform data format and software library for efficiently for storing and exchanging technical data. It supports very large (>2 GB), multiple dimensional data files. An HDF file can contain multiple objects. It may have metadata and notes describing the data, multiple n-dimensional arrays of scalars or data records, images, tables, and user-defined structures. HDF is self-describing and supports compression and optimal tiling of data. It is a highly efficient way to exchange and access "packages" of information including metadata, multi-dimensional data, tables and text.

XML is being used by data farmers to describe model scenarios, studies and other packages of data. HDF is used primarily to store the potentially massive output of the HPC data farming process. The following sections provide an overview of the data exchanged in the flows presented in Figure 2.

2.2 Distillation Outputs (1)

For Data Farming, a modeling or Distillation Scenario Builder application must be able to produce and use three items: a *scenario package,* the model *roadmap*, and the *model*.

The *model* is the complete set of cross-platform software and data that can be placed on a HPC node, executed in a batch mode, and produce a set of end-of-run measurement of effectiveness outputs. The *scenario package* is an XML file and other related data files that defines a set of parameters that can be read by the model and used to instantiate an executable run of the model. The *roadmap* is an XML file that provides a complete description of the parameters that might be in the scenario file and data farmed.

In the case of the Project Albert models Mana (Stephen 2001) and Pythagoras (Bitinas 2002), the *model* consists primarily of a batch version of the applications which are pre-installed on the nodes of HPC system. In more complex modeling systems (such as the Army's Combat XXI model) the model may consist of a set of data and databases.

The *scenario package* consists of an XML file of parameters that can be data farmed and other data files required by the model to execute the scenario. A sample of portions of an XML scenario file for Mana is shown is Figure 3. Mana may also require an elevation file consisting of a BMP formatted image representing a grid of surface elevations for the scenario and a terrain file consisting of a BMP formatted image that represents other attributes of the scenarios environment. Pythagoras currently only inputs the XML scenario file.

```
<Battlefield>
  <x_range> 200 </x_range>
  <y_range> 200 </y_range>
  <LowRealY> 0.0 </LowRealY>
  <HighRealY> 199.0 </HighRealY>
  <LowRealX> 0.0 </LowRealX>
  <HighRealX> 199.0 </HighRealX>
  <SAMode> Map Location </SAMode>
  <LOSMode> Simple </LOSMode>
   <UseNonStdTerrains> No </UseNonStdTerrains>
  <ContactRad> 6.01 </ContactRad>
  <TerrainRange> 5 </TerrainRange>
  <NumSquads> 2 </NumSquads>
  <image type="map" name="terrain.bmp"/>
  <TimeStep> 0 </TimeStep>
  <MaxSteps> 1500 </MaxSteps>
  <settings>
      .....
  </settings>
</Battlefield>

<Squad>
  <SquadType> Trooper </SquadType>
  <SquadActive> Yes </SquadActive>
  <SquadCommsEnable> Yes </SquadCommsEnable>
  <index> 1 </index>
  <SquadName> Blue </SquadName>
  <NumAgents> 100 </NumAgents>
  <SquadOnly> Yes </SquadOnly>
  <Use_Momentum> No </Use_Momentum>
  <Use_Diag> Yes </Use_Diag>
  <MultiOccupy> No </MultiOccupy>
  <AutoRflAgt> No </AutoRflAgt>
  <Use_Going> Yes </Use_Going>
  <Move_Together> No </Move_Together>
  <ResOrgUnknown> No </ResOrgUnknown>
  <ResInorgUnknown> No </ResInorgUnknown>
  <MoveSelectType> Precision </MoveSelectType>
  <Homes>
    <HomeNumberOf> 1 </HomeNumberOf>
    <HomeDistribution> Turns </HomeDistribution>
    <HomePos>
      <Home_x_coord> 189 </Home_x_coord>
      <Home_y_coord> 86 </Home_y_coord>
      <HomeWid> 8 </HomeWid>
      <HomeHgt> 10 </HomeHgt>
      <HomeExclDist> 0 </HomeExclDist>
      <Waypoints>
        <numberOf> 2 </numberOf>
        <loop> No </loop>
        <waypoint>
          <x_coord> 6 </x_coord>
          <y_coord> 89 </y_coord>
        </waypoint>
        <waypoint>
          <x_coord> 129 </x_coord>
          <y_coord> 87 </y_coord>
        </waypoint>
```

Figure 3: A Mana Scenario File Segment

The *roadmap* file is currently produced by hand for Data Farming with Pythagoras, Mana and Combat XXI. Future Data Farming development would be enhance by a models ability to automatically produce roadmap data. Figure 4 is an example of a roadmap file that was hand generated for Mana.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- provides all the farmable parameters
for MANA 3.0
$Revision: 1.2 $ $Date: 2004/07/27 22:01:28 $
-->

<Parameters>
  <Identification>
    <ModelName>Mana</ModelName>
    <ModelVersion>3</ModelVersion>
  </Identification>
  <parameter type="squad">
    <name>General Squad Parameters</name>
    <xpathFindObjectPath>/specification/Squad</xpathFindObje
    <xpathFindIndexPath>index</xpathFindIndexPath>
    <farmable>
      <label>Number of Agents</label>
      <description>Number of Agents in This Squad</descripti
      <type>integer</type>
      <allowedMinValue>1</allowedMinValue>
      <allowedMaxValue>1000</allowedMaxValue>
      <xPath>/specification/Squad[normalize-space(index)='XX
    </farmable>
    <farmable>
      <label>Squad SA Lockout Time</label>
      <description>SA Lockout from Update</description>
      <type>integer</type>
      <allowedMinValue>1</allowedMinValue>
      <allowedMaxValue>1000</allowedMaxValue>
      <xPath>/specification/Squad[normalize-space(index)='XX
    </farmable>
    <farmable>
      <label>Squad Threat Persitence (Organic)</label>
      <description>Number of Steps Threat disappears from th
      <type>integer</type>
      <allowedMinValue>1</allowedMinValue>
      <allowedMaxValue>1000</allowedMaxValue>
      <xPath>/specification/Squad[normalize-space(index)='XX
    </farmable>
```

Figure 4: A Segment of a Mana Roadmap File

2.3 Study Output (2)

A Data Farming Study Building application takes the roadmap data, the scenario package, and other data and user inputs to produce a *study package*. The package consists of the data necessary to execute a Data Farming study in the HPC system.

Either through a graphical user interface, user prepared XML description files, or algorithmically generated XML, a complete description of the basecase scenario, parameter variation excursions, and random seed varied replicates is produced. This is described in the XML study file and is grouped with the scenario package to produce the *study package*.

A Data Farming Study Building application and its resultant study file must be able to support a variety of types and combination of studies. The most basic type of study is the "Gridded" study. A portion of a sample gridded study file is shown in Figure 5. A gridded study consists of a set of excursions, or variations of initial input parameters, that are defined by a start and stop parameter and a fixed "step" value. In the sample in Figure 5, near the bottom, the parameter "instance" of agent is varied from 24 to 48 in steps of 12.

```
  </moeOutput>
  <Version>5</Version>
  <ModelIdentification>
    <ModelName>Pythagoras</ModelName>
    <ModelVersion>
      <Major>1</Major>
      <Minor>7</Minor>
    </ModelVersion>
  </ModelIdentification>
  <StudyIdentification>
    <Name>WP_OFF_Urban01</Name>
    <Description>Gaming example of WP_OFF_Urban</Description>
  </StudyIdentification>
  <UserIdentification>
    <UserName>Piosa, Matthew</UserName>
    <EmailAddress>matthew.piosa@usma.edu</EmailAddress>
    <PhoneNumber>484-225-8545</PhoneNumber>
    <UserID>newid</UserID>
  </UserIdentification>
  <SubmissionParameters>
    <Platform>JEM</Platform>
    <OriginatingMachine>honor.albert.mhpcc.hpc.mil</OriginatingMachine>
  </SubmissionParameters>
  <ModelParameters>
    <RandomGeneratorClass>Default</RandomGeneratorClass>
    <RandomGeneratorMethod>Default</RandomGeneratorMethod>
    <NumberReplicates>3</NumberReplicates>
    <InitialRandomSeed>155029263</InitialRandomSeed>
    <InitialRandomSeed>-541250769</InitialRandomSeed>
    <InitialRandomSeed>737457135</InitialRandomSeed>
    <PlaybacksWanted>no</PlaybacksWanted>
  </ModelParameters>
  <Algorithm type="gridded">
    <ModelRunInformation>
      <ExcursionFileInfo>
        <ExcursionDir>Excursions</ExcursionDir>
        <MOEDir>Output</MOEDir>
        <PlaybackDir>.</PlaybackDir>
        <PlaybackFileStub>viz</PlaybackFileStub>
        <ExcursionFileStub>basecase.xml.</ExcursionFileStub>
        <BasecaseFileName>basecase.xml</BasecaseFileName>
        <MOEFileStub>MOE.</MOEFileStub>
        <TotalExcursions>36</TotalExcursions>
      </ExcursionFileInfo>
      <Dimension type="constant">
        <targetElementField>-1</targetElementField>
        <XPath>/pythagoras/agentList/agent[@name='Blue']/instance</XPath>
        <name>Blue::instance</name>
        <MinimumValue>24</MinimumValue>
        <MaximumValue>48</MaximumValue>
        <DeltaValue>12</DeltaValue>
        <Value>24</Value>
        <Value>36</Value>
        <Value>48</Value>
      </Dimension>
```

Figure 5: A Segment of a Sample Study File

Additional types of studies can be defined in the XML study file. An "enumeration" study requires an XML enumeration file and allows the replacement of entire XML elements and all child elements and attributes with alternate elements defined in the enumeration file. A "lockstep" study is used to link to related model input parame-

ters so that they change consistently with one another. A "DOE" study is produced by applying a rigorous "design of experiments" methodology to produce a more effective selection of parameter variations for execution in a HPC system. Any combination of these study types can be combined into studies optimized for the Data Farmer's needs.

2.4 HPC Outputs (3)

A Data Farming HPC System takes the study package and executes the provided scenario with excursions and replicates as described by the recipe provided in the XML study file. Each execution results in at least one output: an end-of-run measurement of effectiveness (MOE) record. This ASCII text record is either recorded in a file by itself or is concatenated with records from replicates or excursions produced on the same node. These records are normally expected to be in a comma delimited form and further described in the model's roadmap, scenario and study files.

The MOE files from each node are integrated by the HPC system to form input files for Analysis tools. These files currently are produced in one or all of three forms depending on the HPC system being used. These include a standard CSV (comma delimted text) file, a Microsoft Access Database (MDB) file, and an HDF file. The CSV file, being text, can potentially be very large and fairly inefficient to subset, open and manipulate. CSV is a common format and can be opened by many analysis tools. The MDB file can be used with Microsoft Access and other tools, but has limitations in terms of the number of records and fields supported. CSV and MDB files require the use of additional metadata to fully support and understand the data in the files.

HDF output includes the output data from the model, but can also include the study and scenario XML files, as well as any image and other data files used in the Data Farming process. The main advantage of HDF is its ability to package a complete Data Farming study into a efficient single package.

The model running on the node may produce other outputs such as time series data files. These data are often compressed and provided with the MOE output files.

2.5 Analysis Outputs (4)

Analysis tools can use the CSV, MDB or HDF files produced by the HPC system. There is a broad range of Analysis tools being used by Data Farmers. Output from these tools is usually in the form of text-based statistical analysis or data mining output or images of visualization output. Currently these outputs are used by analysts to provide insight or answer questions or to provide direction for additional data farming.

To aid in the analysis of data it is often useful to examine the model runs or runs that produced a specific result. Currently, data farmers can extract the parameters variations, random seeds associated with a specific output and recreate these conditions in the Distillation Scenario Builder to "replay" and observe the execution that produced the results of interest. This is a time consuming and error-prone process. HDF files contain all of the data necessary to automatically generate the initial conditions that produced a give result. An analysis tool using this data should be able to allow users to select a result or set of results and produce the appropriate scenario files for the scenario builder to replay those runs.

2.6 Analysis Inputs (5)

As mentioned earlier, models can produce other forms of output while running in the HPC system or while being used in the GUI-based scenario builder mode. These output are also potential input to analysis tools. Playback data, for example, can consist of unit positions and states as well as weapon or event locations and times, and could be used to closely examine the details of behaviors, paths, and interactions of agents.

Additional time series data is often produced by models include the state of various MOEs at different points of time. In general, all of this time series data is produced in the form of text-based CSV type data. For more complex scenarios, though, multi-object data access can potential provide a much more effective method for playback and data access for this type of data. HDF and relational databases can provide this type of support.

2.7 Generative Analysis Input and Output (6 & 7)

Finally, Data Farming encompasses a process we refer to in figure 2 as "generative analysis." In general this process uses an algorithmic method for examining output from the HPC system and generating new study files for additional data farming.

Inputs for this process are the same as for the analysis tools. An additional input required for this process is some criteria for stopping the iterative process. The criteria might be a maximum number of iterations, a maximum or minimum limit for some output value, observation of an asymptotic limit, or some other algorithm-specific criteria. Outputs for this process can be considered the same as outputs from the study builder.

3 THE WAY AHEAD

Making Data Farming readily available to modelers requires the development of user interfaces and processes that are accessible to individuals not expert in high performance computing systems and coding. Data Farming has been a nascent area of study and the software and tools still require a high level of expertise. Since the component

parts of the Data Farming system are prototypical and are in flux, so are the data interfaces between them.

Component interface requirements may not yet be defined well enough to attempt to *completely* "standardize" data. Establishing flexible prototype standards, shared by the collaborators involved has lead to an understanding of the required processes and their data requirements. With this understanding the next level of data specification can begin. This overview of the data used by Data Farmers is not intended to be exhaustive, but rather a strawman with which to begin discussion of an initial set of base data standards.

ACKNOWLEDGMENTS

Instantiating the idea of Data Farming in the real world of modeling and high performance computing has required the patience, effort and interest of a large number of participants. The authors would like to acknowledge all of the Project Albert collaborators: the modelers, software developers, subject matter experts, analysts, administrative aids, students, teachers, and decision makers from around the world. We are especially grateful for the insight, interest, and time provided by men and women in our armed services.

REFERENCES

Bitinas, Edmund. 2002. Pythagoras: The newest member of the Project Albert family. *Computing Advances in Military OR – WG 31. 70th Military Operations Research Society Symposium*, Ft. Levenworth, June 18 – 20.

Brandstein, A. and Horne, G. 1998. Data Farming: A meta-technique for research in the 21st century. *Maneuver Warfare Science 1998*, Marine Corps Combat Development Command Publication, Quantico, Virginia.

HDF Development Group. 2005. NCSA HDF Home Page. http://hdf.ncsa.uiuc.edu/.

Horne, G. 2001. Beyond point estimates: Operational synthesis and data farming. *Maneuver Warfare Science 2001*. United States Marine Corps Project Albert. Quantico, Virginia.

Horne, G. and Meyer, T. 2004. Data farming: Discovering surprise. *Proceedings of the 2004 Winter Simulation Conference*. Denver, Colorado.

Roger T. Stephen, Mark A. Anderson, Michael K. Lauren. 2002. MANA Map Aware Non-uniform Automata Version 2.0 User's Manual.

W3C. 2003. Extensible Markup Language (XML). http://www.w3.org/XML/.

AUTHOR BIOGRAPHIES

GARY E. HORNE is the founding member of the Data Farming Team at Referentia Systems Inc. He holds a B.S in Mathematics and an M.Ed. in Education from the University of Maryland and a D.Sc. in Operations Research from the George Washington University. He is the originator of the Data Farming methodology and the Director of Project Albert. His email address is ghorne@referentia.com.

TED MEYER is currently working for the MITRE Corporation as the Information Architect for Project Albert. He is a doctoral candidate in George Mason University's Computational Science and Informatics program. Previously, he was the CTO and product designer for Fortner Software LLC, where he managed the development of the company's visualization tools. From 1990 to 1996, Mr. Meyer worked as the Information Architect for NASA's Earth Science Data and Information System Project. Before NASA, he was a Geodesist and Physical Scientist at the Defense Mapping Agency. Mr. Meyer co-authored *Number by Colors: A Guide to Using Color to Understand Technical Data.* His email address is tedmeyer@mitre.org.