Hiveware Terminology

The following are defined terms. Defined terms begin with a capital letter and are in bold.

a. **Ancestor Node** is the Licensor **Node** that created or has acquired Licensor ownership of the Node in question to whose **Hive** revenue is remitted according to their agreement's Compensation Terms.

b. **Author** refers to an **Authorized Participant** who, through the use of a computer device, may edit the **DTD** content model (see **SGML** standard for definition) and associated **Node** code. **Subscriber**s are not part of the **Author Group**.

c. **Author Group** refers to a group of **Author**s to whom an **SGML IDC** has been distributed.

d. **Authorized Participant** is an individual that has accepted terms of an agreement that contains the Compensation Terms, and is an **Editor Type Participant** or a **Subscriber**.

e. **Backup Without Recopying** refers to Hiveware's inherent ability to preserve data content without scheduling periodic recopying of data to another machine. Since Hiveware eliminates servers, data is kept both at the author's site as well as its observers. Over 3 sites provides equal or better protection than scheduled recopying.

f. **Connecting Node** refers to a **Node** in another **Hive** to which this **Node** is a **Subscriber**.

g. **Content** is exemplified by the text or graphics that are associated with a **Context Node**. Linguistically, **Content** is known as the set of signs – text or graphic – that relate the **Context** to the **Content** presentation layer. A **Context Node** with the name Fido could be linguistically represented by the letters F-i-d-o or possibly as a picture, for example:  . Any linguistic sign is **Content** too like, for example, a button click or mouse move.

h. **Content Author** refers to an **Editor Type Participant** that is hierarchically subordinate to the **Root Context Author,** and who has no subordinate child **Node**s, and who has only the capability to change document **Content**.

i. **Content Author Node** refers to the deployed, implemented, and loaded code (e.g., dll for windows, .so of unix) which is owned and used by its **Content Author**.

j. **Context** refers to the a priori environment within which written or spoken natural language **Content** is produced. This environment is implicitly structured which is referred to herein as a semantic grammar or hierarchy. **SGML** highlights this semantic hierarchy as being the best way to represent explicitly this implicit grammar and thereby make it computer tractable. **Context** is also referred to linguistically as deep structure which in this case is the semantic, grammatical

hierarchy of the **Hiveware**® Application. For example, the grammar *Animal : Dog* & Cat*; Dog : MyDog & OtherDogs*; MyDog : Fido* is a semantic grammar at a particular point in time. The **SGML IDC** allows the **SGML DTD** to evolve over time. In other words, *Fido* could be changed to *Woolfy*.

k. **Context Author** refers to an **Editor Type Participant**'s code that is semantically subordinate to the **Root Context Author** and optionally subordinate to other **Context Author**s, and who has the capability to change document **Context** as well as **Content**.

l. **Context Author Node** refers to the deployed, implemented and loaded code owned and used by its **Context Author**.

m. **C++** refers to ISO/IEC 14882:1998 which is a statically-typed computer programming language that **SDCDC Domain Developer (SDD)**s use to develop **Node** executable code. The effect of which is to make **C++** dynamically typed.

n. **Descendent Node** refers to any Sub-Node that is subordinate to a given **Node**.

o. **Document Type Declaration** (**DTD**) refers to **SGML** standard's meta-grammar rules for producing document syntaxes. These produced document syntaxes have the potential of being human readable and therefore able to make natural language sense to the document creator. **SGML** ensures that the produced document syntaxes are computer tractable thus guaranteeing a tie between natural language and its representation as **Context** in a computer.

p. **Dynamic Configurability** is versioning in Hiveware. Each Hiveware Application consists of the basic hiveware executable with a series of libraries that are dynamically linked into it. It is this dynamic linking which gives it its behavior. In contrast, versioned software applications have to make code changes, recompile the code and deliver the new executable to the end user. Patches are an unstructured versions of hiveware's capability.

q. **Editor Type Participant** is a **Root Context Author**, **Context Author** or a **Content Author**.

r. **Licensee Software Developer** is an entity that desires to use the **Licensed Software** in order to develop and distribute a software application to **Authorized Participants** under the provisions of Section 5 herein.

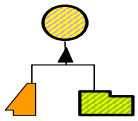
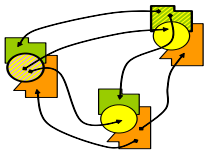
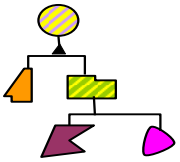
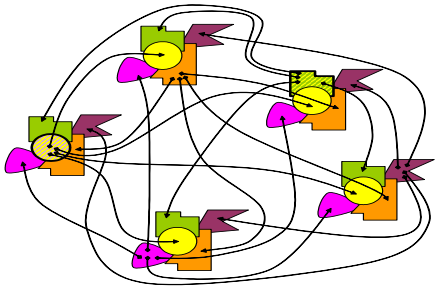
s. **Hive** refers to all of the **Authorized Participant**s running a **Hiveware Application** whose **Root Context Author Node** is the same. **Hive** stands for Hyperstructured Interactive Virtual Environment.

t. **Hive Rules** refer to any set of corporate or group dynamics policies, rules or group laws that ensure the consistency and coherency of the **Node** names and their code behavior. By default, Robert's Rule of Order are in effect, but may be replaced by any set of rules.

u. **Hiveware** stands for *hyperstructured interactive virtual environment soft*ware and is the brand name for the **SDCDC Technology**.

COPYRIGHT (c) 2009-13 BY Hiveware Inc ALL RIGHTS RESERVED.

v. **Hiveware Application** refers to group application software produced using the **SDCDC Te**chnology's SGML IDC as the development platform. Hiveware belongs to the categories of social **Context** and user-generated software. All **Hiveware Application**s function in the following abstract manner described pictorially below:

<table>
<tr>
<td> 
(a. 3 node hive)</td>
<td> 
(b. each node makes one content change)</td>
</tr>
<tr>
<td> 
(c. the  node makes a context change and becomes a 5 node hive)</td>
<td> 
(d. each node makes one content change)</td>
</tr>
</table>

For discussion's sake, let the **Root Context Author** in (a.) ,  , be the leader of a UN workgroup in Darfur, Sudan.  has the topic name *DarfurVillageDestructionMapping*.  is a **Content Author** whose topic is *GPSInputPoint* and  is a **Context Author** whose topic is *PotentialCausalConstructs*. The 3 **Content** changes (b.) are:  writes up the group's mission statement;  reports the first GPS village position, and  makes a **Context** change. He has been thinking of 5 sub topics: *RelationshipToRoads*, *TribalEthnicity*, *ReligiousPersuasion*, *DistanceFromMilitaryCamps* and *DistanceFromBorders*, but since he only has two personnel, a cleric and an ethnographer, to which he can deploy the **Node**s, he creates  , *ReligiousPersuasion* and  , *TribalEthnicity* and deploys these to them. The software showing these **Node**s is changed so that a map shows not only the GPS position of the village, but their religious persuasion and tribal ethnicity. (d.) shows 5 **Content** changes:  adds the new intention of tracking religious persuasion and tribal ethnicity to his mission statement,  has moved to a new village and sends a new GPS point out,  and  , who are now accompanying  make their determinations about the village, and  corrects a bug in the new GPS position/religious persuasion/tribal ethnicity map implementation. The old village still shows only the GPS position while the new village shows all three pieces of information. The next event (not shown) is, 300 new people around the world subscribe to *DarfurVillageDestructionMapping* and

COPYRIGHT (c) 2009-13 BY Hiveware Inc ALL RIGHTS RESERVED.

get populated with their map. The number of **Content** updates for  ,  and  at the next village will then be 3 x (4 + 300) to the 305 replicates. Subsequent additions and changes to the map by any of the **Author**s are automatically distributed (i.e., pushed) to all parties, with no action required on their part.

w. **Hive Related Revenue Percentage** refers to the proportion of revenue due the **Licensor's Hive** made from the continual sale of the **Target Work** to subscribers to his hive.

x. **Hiveware Domain Designer (HDD)** refers to a **Context Author** or **Root Context Author Node** who designs or configures an already developed **Node** implementation.

y. **Licensed Software** refers to the software created by Licensor in executable form, together with any documentation supplied by Licensor to Licensee under this License Agreement.

z. **Licensor** means Hiveware Inc or any **Root Context Author** Licensee.

aa. **Licensor's Hive** refers to the hive of the **Ancestor Node** that signed the Licensee up.

bb. **Node** refers to a computer address of an **Author** and is a semantic element identifiable in the **SGML IDC**'s semantic grammar **DTD**. A **Node** comprises named, implemented and loadable code that was created specifically to perform operations that are specific to and consistent with the semantic category it represents. A **Node** is therefore a duality of representation and operation.

cc. **Patent and Trademark Assets** refers to all of Hiveware Inc's exclusively licensed patents and trademarks. These patents are: U.S. Patent No. 7,124,362, European Application No. 02759443.1, Australian Patent No. 2002324778, Chinese Patent Application No. 02816585.3, Hong Kong Patent Application No. 04107219.6, Canadian Patent Application No. 2,458,860, Indian Patent Application No. 00192/DELNP/2004, and Philippine Patent No. 1-2004-500256, US Registered Trademarks SDCDC® Serial No. 78328626 and HIVEWARE® Serial No. 77020344 (hereinafter the SDCDC Patents and Trademarks), that is directed to methods for enabling cooperative software applications with the exception of this and other Licensor rights which have been conveyed with this agreement to their Licensor.

dd. **Replicate NodeTree** refers to each site's copy of all the continually updated **Subscriber Node**s plus **Editor Type Participant Node**s.

ee. **Root Context Author** refers to an **Editor Type Participant** that represents the upper most **Node** in a document's semantic hierarchy.

ff. **Root Context Generator** refers to the **Editor Type Participant** capability of being able to create new and deploy new **Hiveware Application**s beginning with the creation of the top most **Node**.

gg. **SDCDC Patents** refers to U.S. Patent No. 7,124,362, and any inventions described and claimed therein and continued prosecution applications claiming

COPYRIGHT (c) 2009-13 BY Hiveware Inc ALL RIGHTS RESERVED.

priority thereto to the extent the claims are directed to subject matter disclosed in such patents and patent applications, and any patents issuing thereon or reissues, reexaminations or extensions thereof, and any and all corresponding foreign patents, and patent applications or foreign counterparts. The **SDCDC Patents** relate to an **SGML IDC** and development environment.

hh. **SDCDC Domain Developer (SDD)** refers to a **Context Author** or a **Root Context Author Node** who develops **C++** implementation designed to be useable and inheritable by other **Node**s.

ii. **SDCDC Technology** refers to the Synchronous Distributed Context Distributed Content technology disclosed in the **SDCDC Patents**.

jj. **SGML** refers to ISO 8879 standard and stands for Standard Generalized Markup Language.

kk. **SGML IDC** refers to subject matter of the **SDCDC Patents** that is a **Root Context Generator** and which stands for **SGML** Interactive Distributed Compiler.

ll. **Subscriber** refers to an individual (person or entity) that has received an executable that is part of a **Hiveware Application** and is permitted to view or experience a product of that executable subject to the Compensation Terms.

mm. **Sub-Node** refers to a **Node** whose semantic **Content** is subordinate to another **Node**.

nn. Relating the terms: refer to the paper entitled "Explanation of the HIVEWARE Architecture" at www.hiveware.com under 'for IT professionals' for additional explanation of the terms above.

oo. **Target Work** refers to the sale of access to a hive's target work is a continual process because the information in the target work is continually updated by the hive's authors. A Target Work may be a traditional document or compound document, or it may be primarily graphical in nature.

COPYRIGHT (c) 2009-13 BY Hiveware Inc ALL RIGHTS RESERVED.