Project ref. no.	IST-2000-26095
Project title	NITE: Natural Interactivity Tools Engineering

Deliverable status	Public
Contractual date of delivery	31 July 2003
Actual date of delivery	30 September 2003
Deliverable number	D6.3
Deliverable title	NITE Final Report
Type	Report
Status & version	Final
Number of pages	37
WP contributing to the deliverable	WP6
WP / Task responsible	NISLab
Author(s)	Niels Ole Bernsen and Laila Dybkjær
EC Project Officer	Philippe Gelin
Keywords	NITE project achievements, future prospects
Abstract (for dissemination)	This deliverable summarises the NITE project and presents the results achieved in the project. It also discusses future prospects.

Final Report

Natural Interactivity Tools Engineering

NITE - http/nite.sdu.dk

30 September 2003

Authors

Niels Ole Bernsen[1] and Laila Dybkjær[1]

1: NISLab, Odense, Denmark

Table of Contents

1 Executive summary

The NITE project (http://nite.nis.sdu.dk) was launched in March 2001 in response to the – partly perceived, partly anticipated – strongly increasing need for standards and tools in support of creating, annotating, evaluating and exploiting natural interactivity and multimodal (NIMM) data resources. Not only is the production of value-added NIMM corpus data time-consuming and costly, so that re-use of annotated data would seem very attractive. The inherent complexity of coordinated human communicative behaviour, involving speech, gesture, gaze, facial expression, body posture, and use of all manner of objects as an integral part of the communication, is so huge that the only way of exploring it efficiently would seem to be through the use of general-purpose NIMM annotation tools. Exploration through annotation-by-hand is only feasible in principle and is would seem quite unlikely that the research communities involved would manage, within any reasonable time-frame, to build all the special-purpose tools needed in the field, each taking good care of annotation of a particular kind of NIMM behaviour.

So, the objective of NITE has been to (i) carry out the research necessary for (ii) building (a) best practice, general-purpose toolset(s) for multi-level, cross-level, and cross-modality annotation, retrieval, and exploitation of multi-party natural interactive human-human and human-machine dialogue data. In this endeavour, NITE has been able to build on research results from the ISLE (International Standards for Language Engineering, 2000-2002, http://www.ilc.cnr.it/EAGLES96/home.html) Working Group on Natural Interactivity and Multimodality (NIMM, http://isle.nis.sdu.dk). ISLE NIMM has produced a comprehensive map of the state of the art world-wide in NIMM data resources, coding schemes, and coding tools, contributed to the specification of best practices in these fields, addressed the emerging issue of meta-data information, and even produced a first draft requirements specification of a general-purpose NIMM coding tool. In addition, NITE has been able to build on the results of the MATE (Multi-level Annotation Tools Engineering, http://mate.nis.sdu.dk) project (1998-1999) which built a toolset for multi-level and cross-level annotation of spoken dialogue data. In fact, NITE not only builds on the MATE technology but also, to some extent, on the limited-purpose NIMM annotation technologies represented by The Observer and ANVIL.

Based on the scientific and technical starting-points just mentioned, NITE has ventured into the unknown territory of general-purpose NIMM annotation tools and has come out with a number of usable tools covering most aspects of the NITE objective. Three NITE development strands have been pursued. One strand, the NITE WorkBench for Windows, or NWB, is based on a Windows platform, and aims at users who want an easy-to-use interface that requires no programming skills. A second strand, the NITE XML Toolkit, or NXT, is cross-platform and builds on MATE and ANVIL. This strand focuses on users who are able and willing to do some programming to use the tool. The third strand has enabled, and will enable, the commercial The Observer tool to increasingly support some amount of annotation of natural interactive communication. In addition, the NITE consortium has gained unique knowledge of the theoretical and practical challenges facing a truly general-purpose NIMM coding tool. The consortium members expect to continue their efforts with sufficient momentum to be able to offer to the world the first tools of this kind. Meanwhile, our original projections concerning the need for general-purpose NIMM annotation tools have been amply confirmed by research developments in the past 2-3 years.

2 Project timetable

NITE started on 1 April 2001. At the project kick-off meeting, the consortium decided to pursue three development strands instead of the planned two strands, i.e. the NITE toolset and The Observer. The basic idea was to develop for three target user groups, i.e. programmers, non-programmer users who want a full general-purpose NIMM coding tool, and future users of The Observer who want a version of the tool which includes some aspects of the common NITE tool specification (see Section 4.1). Compared to the original NITE concept, the new, added target user group was that of non-programmer users who want a full general-purpose NIMM coding tool. This decision clearly implied an even more ambitious project plan than the one in the NITE contract but it was felt that it was very much worth the increased effort to also aim to cater for the potentially very large group of “ordinary” users.

task	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
WP1	—	—	—	—	—	—																						
WP2	—	—	—	—	—	—	—	—	—	—	—	—																
WP3				—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	
WP4							—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
WP5	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
WP6	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	01	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**
												M1																**M2**
	W1					**W2**						**W3**						**W4**							**W5**			

Table 2.1. Duration of work packages and tasks included in the work packages. Month 12 and month 28 were concluded with a milestone. Five project workshops were held. Two additional software developers' workshops were held for the smooth development of the distributed software development part of the project. Project workshops were spread as evenly as possible. Open workshops took place in connection with major relevant conferences or workshops.

In organisational terms, the three-strand decision meant that the NITE project sub-divided into three sub-groups as far as tool development was concerned. Importantly, the three groups would still develop a common NITE tools specification together, i.e. the one presented in D1.1 and the addendum to D1.1. From there, the groups would adhere to the specification in significantly different ways. The first (NXT) sub-group pursued the MATE-based agenda of developing a toolset for users who could program their own coding schemes and the corresponding coding file visualisations. This group consisted of U. Edinburgh, DFKI and IMS as developer sites, Edinburgh doing the core XML engine, DFKI doing various plug-ins in collaboration with their own in-house Anvil coding tool project, and IMS doing the query functionality. The second (NWB) sub-group consisted of NISLab which were to develop a general-purpose coding tool for “ordinary users”. The third sub-group consisted of the Noldus company which were to include selected common NITE requirements into their specifications of future versions of The Observer. Noldus stepped down from their planned role as development leaders in NITE more generally. Edinburgh assumed leadership of the first-mentioned group’s work and also took over the WP responsibility for delivering the common developers' deliverables. This new division of labour and responsibilities has probably entailed some differences in workload per site over the duration of the project and compared to what was originally planned, but no changes in budget allocation were asked for.

At the first-year review, the reviewers confirmed this new work plan for the project, requesting an addendum to the requirements specification deliverable D1.1. The D1.1 Addendum deliverable was delivered on time following the review. Another consequence of the revised work plan was that the project as a whole needed some more time to complete its work. The Commission kindly granted NITE a four months extension without additional funding, so that work could continue after the contractual deadline by end March 2003 and until the end of July 2003.

Like its predecessor project MATE's Advisory Panel (AP), the NITE AP was planned to have a prominent role in NITE throughout. This role may, in fact, have been slightly less prominent than in MATE, for the following reason. We discovered that we all agreed that MATE had somewhat "oversold" its message from project Day 1. The result was, certainly, a large number of AP member sites who took a highly constructive role in the project. However, they also suffered from the fact that the MATE project took quite long to deliver anything which could actually be used, and the carefully crafted MATE questionnaire had only a limited response at the end of the project. Thus, in NITE, we adopted a rather cautious approach to AP advertising and recruitment, the result being that NITE has had a smaller, some 35 members strong, AP. However, as NITE has actually been able to deliver several software releases for user testing in the life-time of the project, the end result has been that NITE has seen very active AP involvement in the final 9 months of the project. Evaluation results are described in D4.1 and D.4.2 and commented on in Section 5 below.

3 Theoretical achievements

3.1 General findings

It is important to note that the field of entirely general-purpose NIMM coding has hardly been theoretically explored at all. In the research literature, apart from what has been produced in ISLE and NITE, no detailed analyses can be found of the workflow of NIMM data coders in general, no general analysis can be found of the types of coding needed, and no general analysis can be found of the types of data and coding file visualisation needed by coders. In this context, NITE has benefited a great deal from the ISLE surveys of existing NIMM data resources, coding schemes, and coding tools.

In NITE, we adopted from early on a guiding use case which was the one of coding, at all levels required and in all modalities involved, turn-taking cues in human-human conversation. This use case appeared to illustrate well the complexity facing a general-purpose NIMM coding tool, both in terms of the coding schemes involved and in terms of the complexity of raw data and coding file visualisation needed for coders to accomplish their tasks (cf. D1.1). Some important findings made in studying this use case and others are described in this section by reference to illustrations shown below.

3.1.1 Timestamped and structure coding

In timestamped coding, a timestamp (start and end time, a time point) is assigned to a phenomenon in the data. In structure coding, a tag is assigned to one or more phenomena which have been timestamped already, such as in POS-tagging in Table 3.1. Both types of coding must be supported by a general-purpose NIMM coding tool.

Table 3.1 shows a medium-complexity example of the communicative behaviour coding needed for developing future natural interactive systems. The table shows an interface design sketch (analogue view, see below) of the coding of (part of) the spoken utterance and gesture parts of a single communicative act at a number of coding levels, i.e., from the top down: POS (part of speech, POS1.1-1.4), semantic (SE1), gesture-speech coordination (C1), co-reference (CoR1), word-level segmentation (WL1), prosodic level (Tobi), gesture level (gesture type (GC1), gesture phases (GPC1), both for right (R) and left (L) arm/hand), utterance level (UL1), and discourse (DLC1), all referring to the common timeline (bottom). Coding at each level makes use of a particular coding scheme (not shown in Table 3.1). For brevity, only a single human communicator (S1) is shown.

3.1.2 Symbolic and analogue visualisation

Symbolic visualisation, e.g. in table format (cf. Figure 4.3.1.3), enables visualisation of a significant fragment of the coded communication, as needed in, e.g., utterance and discourse-level analysis. This is not possible in analogue visualisation, cf. Table 3.1 which only shows a single utterance. Analogue visualisation, on the other hand, enables immediate perception of temporal relationships in the data. It appears that a general-purpose NIMM coding tool should enable both kinds of visualisation.

3.1.3 Formal data properties

Across modalities and coding levels, it is possible to identify more abstract properties in the data, such as the within-level long-range dependencies to be found in, e.g., co-reference relations (cf. Table 3.1), or in multimodal coordination, an example of which is shown in Table 3.1. It is crucial to the successful development of a general-purpose NIMM coding tool that the tool takes into account as many as possible of the formal properties in the data to be coded.

CS/time									
POS1.1-S1	S								SC
POS1.2-S1			VP hlem=buy						SC
SE1-S1	frame=buy, fo=agt			frame=buy, fo=pat			frame=buy, fo=ben		SC
POS1.3-S1	NP hlem=man			NP hlem=toy		PP prep=for hlem=child	NP hlem=child		SC
POS1.4-S1	DT	NN	VBD	DT	NNS	IN	PPS	NNS	SC
C1-S1				c2				c3	SC
CoR1-S1	anchor5						anaphor5		SC
WL1-S1	w30 the	w31 man	w32 bought	w33 these	w34 toys	w35 for	w36 his	w37. children	TC
Tobi-S1				H*				H*L	SC
GPC1-S1-R		prep	stroke	hold		retract			TC
GPC1-S1-L						N/A			TC
GC1-S1-R		g3-de, target = toys							TC
GC1-S1-L						g4-di			TC
UL1-S1	U5. the man bought these toys for his children								TC
DLC1	main								TC
time	....								N/A

Table 3.1. Analogue view of semi-complex natural interactivity coding. The left-hand column refers to the coding schemes (CSs) used. Sn is speaker n. The right-most column shows if the coding is structural (SC) or timestamped (TC), see Section 3.1.1.

3.1.4 Visualising cross-level and cross-modality links

In future complex NIMM data coding, it will be essential for tool users to be able to visualise, and possibly store, cross-level and cross-modality links between phenomena in the data which form part of coordinated natural interactive communication. This challenge is one which still needs to find appropriate design solutions in a general-purpose NIMM coding tool.

3.2 Other theoretical results

In the NITE project we have developed a markup framework for annotation at multiple levels and across levels and modalities and we have analysed existing coding schemes for markup of natural interactivity data and identified best practice in the area. This work is presented in the following sections.

3.2.1 The NITE markup framework

The developed NITE markup framework was an extension of the markup framework developed in the MATE project [Dybkjær et al. 1998] which covered spoken dialogue data but not full natural interactivity data. The NITE markup framework is described in deliverable D2.2. This report discusses, and proposes solutions to, the general issues involved in creating a general standard for the markup of natural interactivity corpora. It presents the NITE markup framework including raw data, coding modules and coding schemes, coding files, and meta-data, and it describes important issues to be addressed. In addition, the report presents the proposed realisation of the markup framework in two of the three software strands in the NITE project, i.e. the NITE WorkBench, NWB, and the NITE XML Toolkit, NXT, respectively.

The proposed NITE markup framework file structure is shown in Figure 3.2.1.1. The basic annotation process requires a coding module and either raw data (audio and/or video) or an already annotated coding file referencing raw data. A coding module consists of a coding scheme and meta-data information. When applied to raw data or to an existing coding file, the coding scheme produces a new coding file. Coding file meta-data references the meta-data of the coding module. Coding file meta-data may also directly reference the meta-data of the raw data. Otherwise, raw data reference is made indirectly via reference to the meta-data of another coding file. Links are also made to the meta-data of other coding files if these are referenced in the coding file.

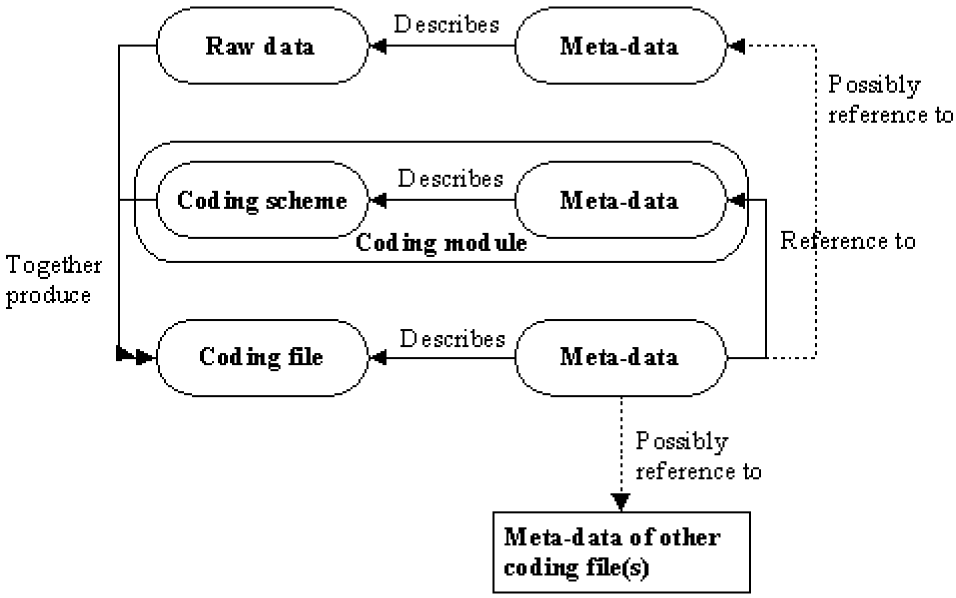

Figure 3.2.1.1: General NITE markup framework file structure.

Raw data considered in NITE includes audio files, video files, images, logs, data files, notes, and transcription files in case no audio raw data is available.

Meta-data is a research issue of its own. NITE has taken a pragmatic approach by suggesting for raw data meta-data, coding scheme meta-data, and coding file meta-data, respectively, a set of information entries which one at least should consider to fill in. Whether a particular piece of information is relevant may in some cases depend e.g. on the kind of raw data. For example camera information is of course not relevant if the raw data is an audio file.

The coding module is a central concept in the markup framework. A coding module includes two types of information. One type is mainly intended for the user and another type is mainly intended for the system. The information intended for the user may be viewed as the coding scheme concepts together with their meta-data information. The information intended for the system basically corresponds to a tagset.

The name of the coding module is intended for both the user and the system.

Information mainly intended for the user includes:

1. author(s)
2. version
3. notes (references to literature, validation information, comments, etc.)
4. purpose of the coding module
5. coding level(s) (e.g. dialogue acts, hand gesture, nose wrinkles, …)
6. description of data source type(s) required for use of the coding module
7. explanation of references to other coding modules
8. coding procedure
9. coding example showing the coding scheme markup in use
10. clear description of each phenomenon, example(s) of each phenomenon. Phenomena such as gestures and facial expressions may be illustrated in static or dynamic images to further help the user identify their occurrence in the data

Information primarily intended for the system includes:

11. a markup declaration, possibly hierarchically ordered, of the tags for the (individually named) phenomena which can be marked up using the coding module
12. coding files referenced

Important issues in natural interactivity coding are also discussed in the report. These include the questions of how to deal with cross-level and cross-modality coding, overlapping interactions, intersecting hierarchies, long-range dependencies, synchronous and asynchronous phenomena, and time alignment. For further details we refer to D2.2

3.2.2 Annotation schemes

Regarding annotation schemes, the NITE project could to quite some extent draw on work already done in the ISLE NIMM (natural interactivity and multimodality) working group (isle.nis.sdu.dk). In [Knudsen et al. 2002], 21 existing coding schemes for markup of gesture and facial expressions possibly in combination with speech were described and reviewed. This report served as point of departure in NITE both for two chapters on existing coding schemes in deliverable D2.1 which also included some more theoretical work on gestures, and for the report on best practice coding schemes, see deliverable D2.3.

Report D2.1 gives an overview of existing gesture, facial expression, and cross-modality coding schemes. The report presents an overview of the foundations of natural interactivity, and summarises and updates the results of the mentioned ISLE survey of gesture and facial expression coding schemes. Regarding cross-modality issues, sequences of events are identified characterised by the co-occurrence of events from different selected modalities. In addition, examples have been collected from various gesture dictionaries including one entry of the Berlin Gesture Dictionary, and an overview of coding schemes for nonverbal behaviour is provided.

Based on this work and on the NITE markup framework described in Section 3.2.1, NITE also did work on identifying best practice gesture, facial expression, and cross-modality coding schemes, see D2.3. This report presents coding scheme recommendations for natural interactivity research as well as coding module specifications of the recommended schemes for possible inclusion in the NITE workbench. The recommendations are based on clear definitions of the notions of coding scheme, meta-schema, coding module etc. Evaluation criteria were devised for all these concepts in order to support the recommendations. For facial expression coding, semi-standards have already developed. Three coding schemes are presented and recommended in the report. For gesture coding, a modular approach is proposed for building a gesture coding scheme to cope with the widely diverging research aims. Two recommendations are presented for the most basic modules. For cross-modality coding, we point to cross-modality aspects in the surveyed schemes and conclude that there are no coding schemes yet that are generic enough to qualify for recommendation. Finally, coding modules in accordance with the markup framework are defined for all three facial expression coding schemes and one recommended gesture coding scheme.

The original intention was to include some best practice coding schemes in the developed tools. However, in NITE, as opposed to MATE, it turned out not to be as important to select best-practice coding schemes for inclusion into the tools. The best practice facial coding schemes, for instance, already come with tool support. Clearly, if there is a tool available which has been tailored to support a specific coding scheme, this tool would often be preferable to a general-purpose one for applying this particular coding scheme. The primary need for general-purpose tools is in the exploration of complex natural interactive and multimodal behaviours, which means that ease of entering, or creating, new coding schemes is essential.

4 Technical Achievements

4.1 Requirements specification

In essence, the ISLE draft requirements specification of a general-purpose NIMM coding tool [Dybkjær et al. 2001a] stated that, basically, the tool should enable the user to:

1. enter any natural interactivity coding scheme, however experimental, into the tool;
2. enter raw data (audio, video) resources in various common formats into the tool;
3. minutely control the raw data (milliseconds, frame-by-frame);
4. annotate (code, tag) the raw data using the entered coding scheme and an automatically generated tag palette for ease and speed of coding;
5. make user cognitive and workflow-compatible visualisation and customisation of the coding files;
6. query the coded data to extract relevant coding file information;
7. possibly do some key statistical analysis of the coded data, such as inter-coder agreement analysis;
8. export the resulting coding files for use by other tools, e.g. using XML.

Software engineering-wise, the tool should have an open architecture for adding new components and functionalities. For instance, plug-ins of various graphical representations of the speech signal and synchronised with it, such as waveform, pitch contour, spectrogram, etc. For ease of maintenance, the tool should embody a clean separation of user interface, application logic, and internal representation (data model).

In addition, the tool should be robust and fast enough to be used for research purposes, also in case of large amounts of data, and it should be user friendly in the sense of being compatible with coders' workflow.

The ISLE draft requirements specification of a general-purpose NIMM coding tool were jointly refined by the NITE consortium and the results are described in D1.1 and D1.1 Addendum.

4.2 Meeting the requirements

At the start of NITE, it was clear that a coding tool meeting the above requirements did not yet exist world-wide by far. In fact, such a tool still does not exist. In order to present a structured view of the present state of the art, we may define the following concepts. A *special-purpose coding tool* is one which supports the coding of NIMM data at a particular level and in a particular modality. Some existing tools do this quite well, such as Transcriber for orthographic transcription and Praat for phonetic transcription. However, these tools are drops in the ocean when facing the challenge of general-purpose NIMM coding. A *limited-purpose tool* is one which aims to support as large a fraction of NIMM coding as possible but which still falls short, more or less, of meeting the requirements above. It may be noted that, relative to those requirements, the shortcomings of a particular limited-purpose NIMM coding tool can be defined quite precisely. A *general-purpose tool,* finally, is one which meets the above requirements head-on. The tool still cannot be guaranteed to satisfy every possible coding purpose, simply because these will remain intractable. Nor can the general-purpose tool be guaranteed to satisfy every possible user in the way it enables graphical presentation of annotation schemes, coding files, query results, etc., also because these will remain intractable.

Based on refinement of the above requirements specification, NITE has produced three limited-purpose toolsets, all of which have the potential of eventually satisfying the requirements specification in full. At the start of NITE, it became clear that the requirements specifications could be satisfied in many different ways. One important distinction had to do with the target user group. If these are relatively computer illiterate, it does not make sense to offer them a toolset which demands significant programming skills. If, on the other hand, the target users are programmers or have easy access to programming support, the manner in which the requirements specification could be satisfied changes considerable. In this case, it might be possible to offer to the users a toolset which, like the MATE Workbench, enables them to program their coding scheme and its on-screen visualisation. Another important distinction also had to do with the target users. Only, this distinction was between users who may be already familiar with a particular coding tool and who want to venture, to some limited extent, into larger-scale NIMM coding, and users who are dedicated to exploring NIMM behaviours to their limits. The former may be satisfied with a limited-purpose version of their familiar tool whereas the latter want a general-purpose tool.

The NITE toolsets well reflect the distinctions made in the preceding paragraph. The NWB is aimed at users without significant programming knowledge who want to do full NIMM data coding. These researchers are experts in data coding and theory but not experts in tools programming. The NXT is aimed at users who have the necessary programming skills and who want to do full NIMM data coding. And The Observer is aimed at users without significant programming knowledge who want to do limited NIMM data coding. Given this 2-by-2 option space, the only user group which NITE has not been aiming at are the users who are excellent programmers but who do not want to explore the field of NIMM behaviours in full! This is hardly a serious limitation to the NITE results.

So, how far did we manage to support full and unconstrained NIMM data coding? By definition, so to speak, The Observer is not going to go all the way. However, this observation is counter-balanced by facts such as that The Observer is professional software with the robustness and maintenance support this entails. For obvious reasons, the NWB and NXT still remain less robust research prototypes. Between them, NWB and NXT are very difficult to compare. Firstly, these tools aim at different target user groups. In a special sense, NXT is very general-purpose already, only you have to do a considerable amount of programming in order to code any particular kind of NIMM behaviour. If you can do that and want to do that, you may get more of less the working environment you want but you have to program again in order to code something else. NWB, on the other hand, is still limited-purpose but you do not have to be a programmer to use it.

It should also be added that, to compare the NITE tools in a more comprehensive and transparent way, far more parameters are needed than mentioned so far. Probably the most comprehensive, parameterised comparison which has been done so far, both among the NITE tools and between these and other current state-of-the-art limited-purpose NIMM coding tools, is the one presented by NISLab NITE workers in August 2003 at the Eurospeech conference in Geneva. This comparison uses the following parameters which, incidentally, illustrate the complex challenges facing general-purpose NIMM coding tools:

1. Overall purpose
2. Providers
3. Platforms
4. Implementation
5. License issues
6. Supported formats
7. Interface

8. Information extraction
9. AnalysisCoding schemes
11. Coding palette
12. Types of coding
13. Coding file view
14. Custom visualisation
15. Import/export
16. Internal data representation
17. Meta-data support
18. Advantages
19. Disadvantages

4.3 The NITE tools

In the terms of the theoretical apparatus introduced in Section 3.1 above, it may be said that the present, third release of the NWB supports timestamped coding of virtually any kind of NIMM data. The NWB also supports structure coding but only in the curtailed sense of enabling timestamped structure coding. The same applies, to some extent, to The Observer which, contrary to the NWB, is limited by its internal data model with respect to the coding schemes it can support. Moreover, The Observer still does not well support the coding of spoken dialogue data. The NXT supports structure coding once the user has programmed what is required.

In terms of the distinction between symbolic and analogue visualisation (Section 3.1), the NWB currently supports symbolic visualisation. Information-equivalent analogue visualisation is a next-step goal. The same holds for The Observer whereas the NXT may support analogue visualisation of particular coding scheme applications provided sufficient programming capability at the coding site. None of the NITE tools have solved the general problem of coordinated behaviour visualisation which remains a challenge in the field. Both the NWB and The Observer offer various customisation options which can help users display coordination links.

In the following, we present the NITE tools through series of screenshots and accompanying text. For obvious reasons, it is not possible to describe or characterise any of the tools in full in the present report. Additional information is provided in the documentation and user manuals for each tool. We refer to the NWB and NXT websites and to The Observer user manuals and documentation.

4.3.1 NWB

The NWB is meant for users who want a ready-for-use tool for doing annotation. It runs a on Windows platform. It is developed in C++ and uses a relational database for data storage. The NWB is available as freeware for users. The following screenshots illustrate part of the functionality provided by the NWB.

Figure 4.3.1.1 shows part of the procedure for entering a new coding scheme. The rightmost window shows how elements are entered into the Transcriber coding scheme selected in the leftmost window. Attributes can be added to elements by using the attribute list button visible at the bottom of the rightmost window.

Figure 4.3.1.2 shows how transcription can be done using NWB.

Figure 4.3.1.3 shows the ordinary annotation view in NWB. At the bottom is the video window and the control board which shows that one can go both by milliseconds and frame by frame in the video. In the middle the speech wave is visible. It is time aligned with the video as marked by the red bar. At the top is the annotation window. It shows start and end times for tags and it shows in the third column all entered tags.

By using the query tool one can get various visualisations of the annotated data. Figure 4.3.1.4 shows the inserted tags divided into separate columns so that one column shows speakers, a second shows the transcribed text, a third one speech deictics, and so on.

Figure 4.3.1.5 shows the query interface which. So far plain SQL is the language used for asking queries.

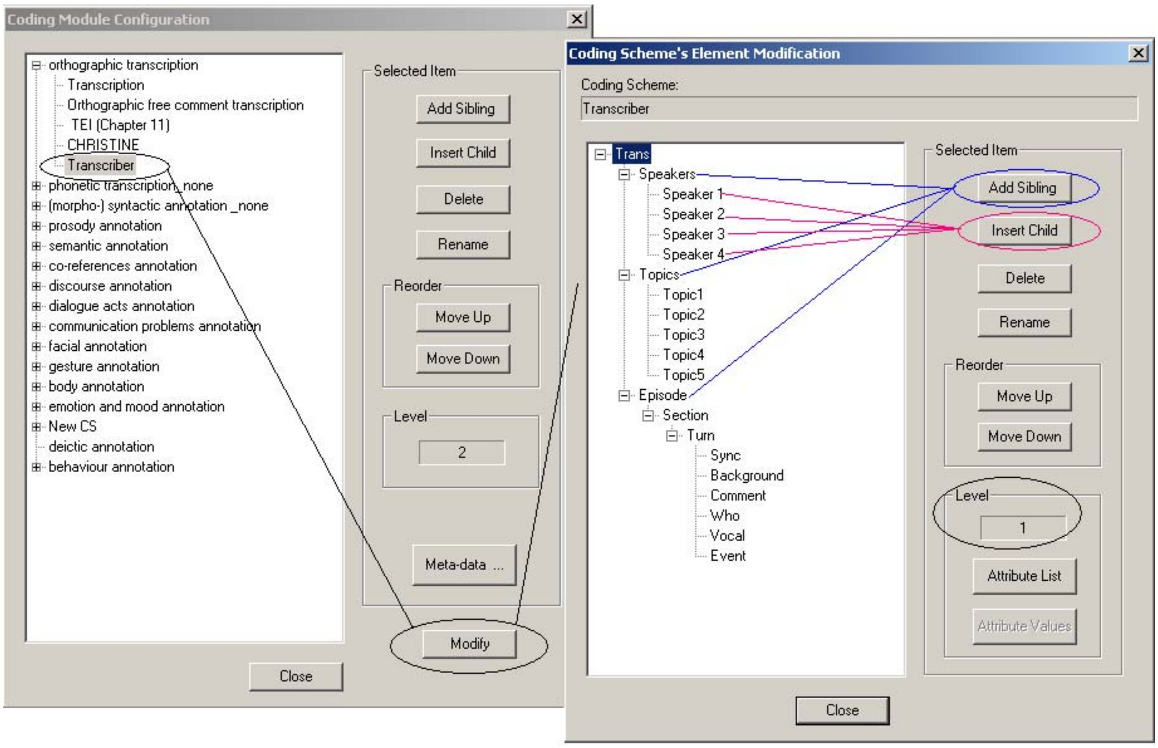

Figure 4.3.1.1: Adding elements in a coding scheme.

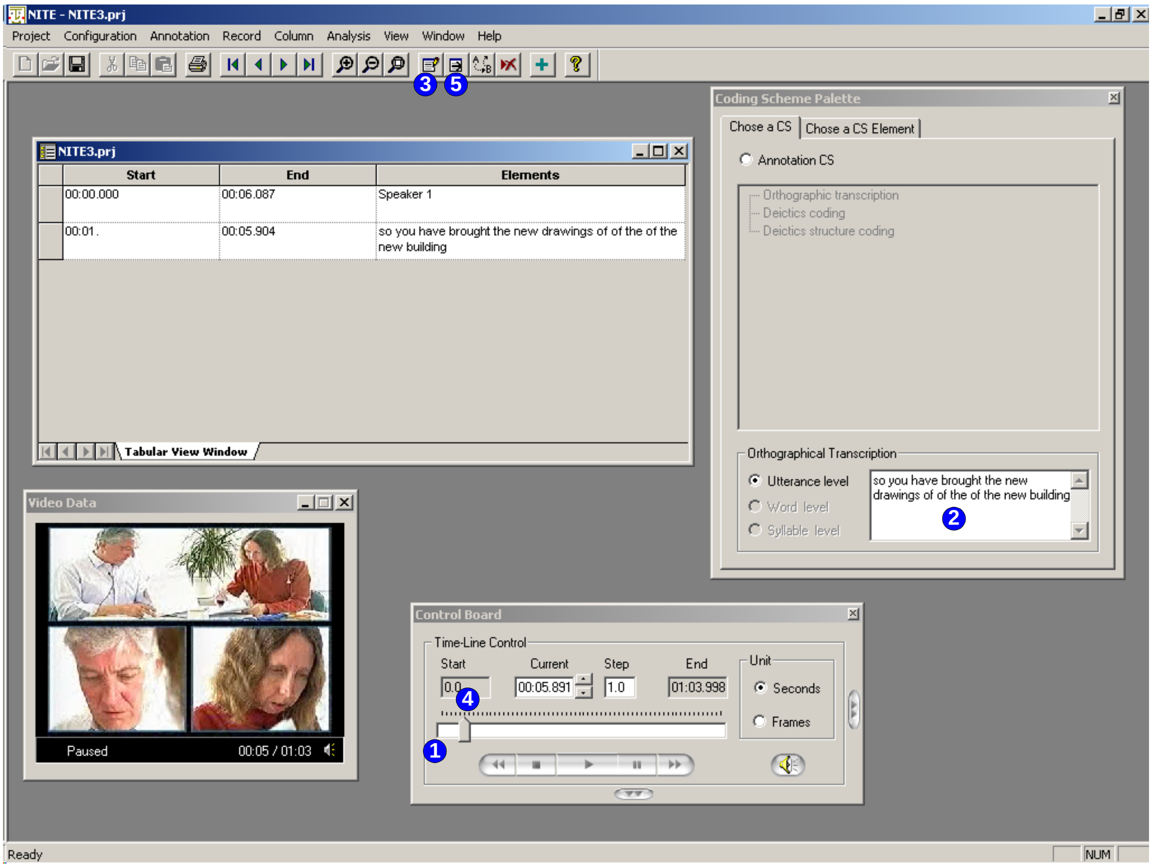

Figure 4.3.1.2: Doing transcription with the NWB.

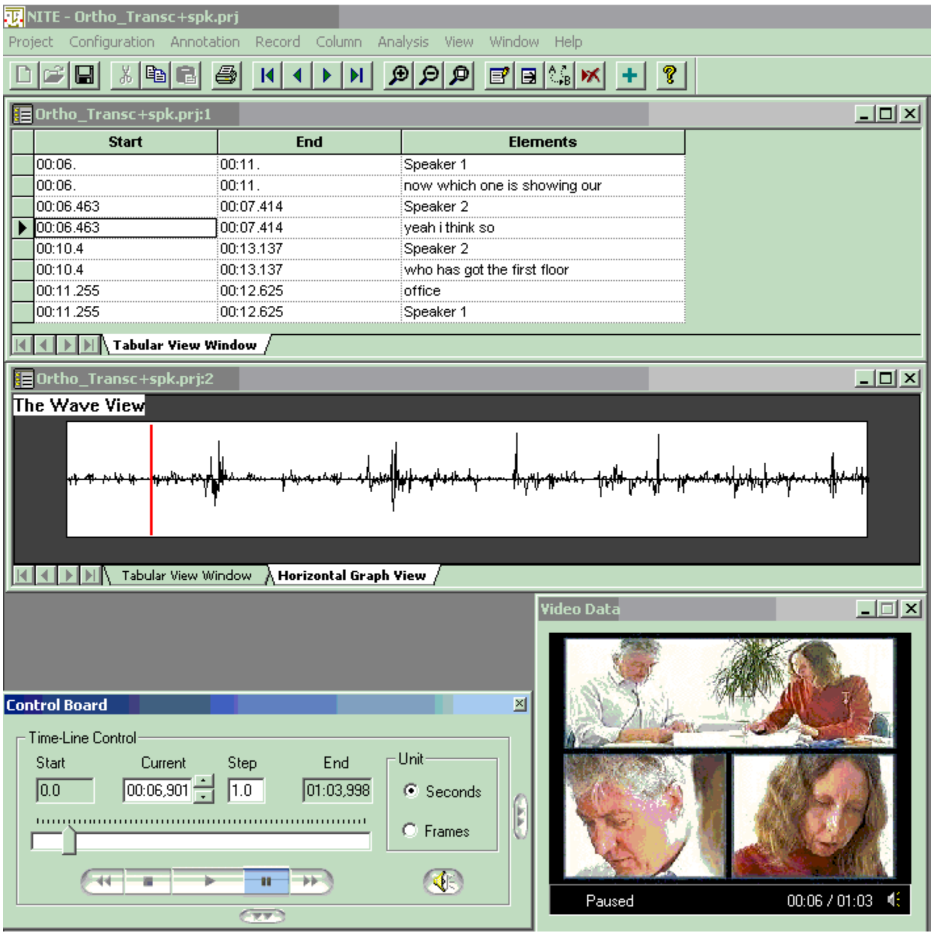

Figure 4.3.1.3: Annotation view in NWB.

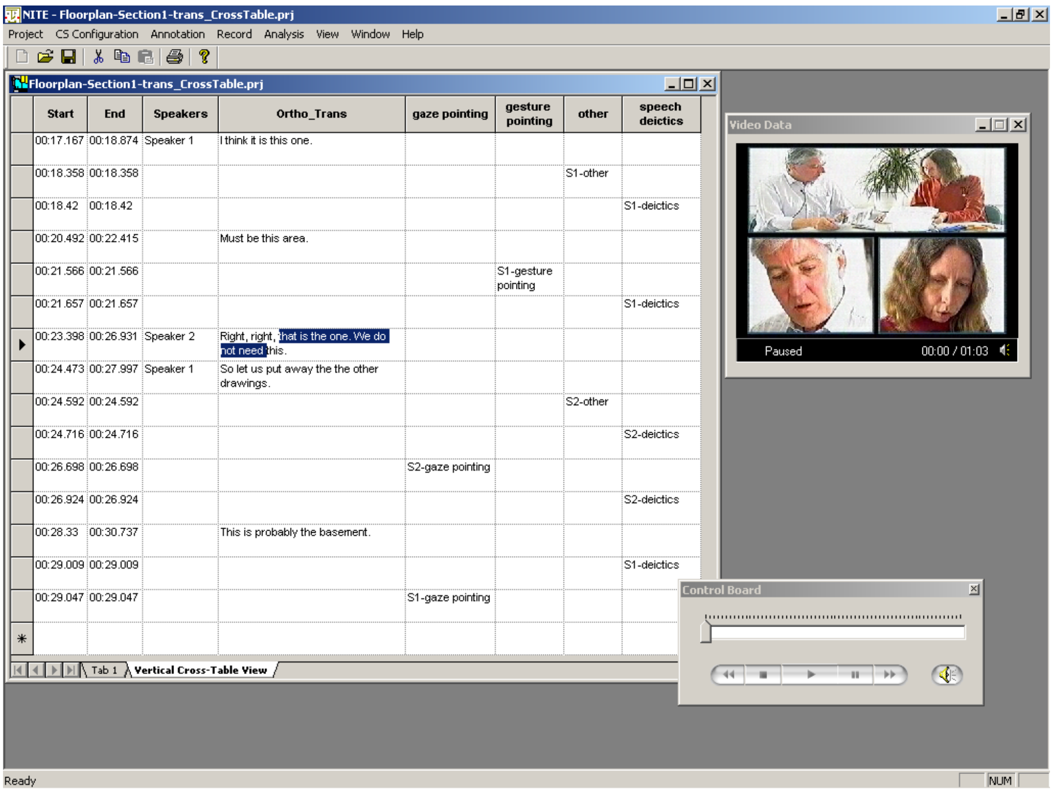

Figure 4.3.1.4: Visualisation of coded data.

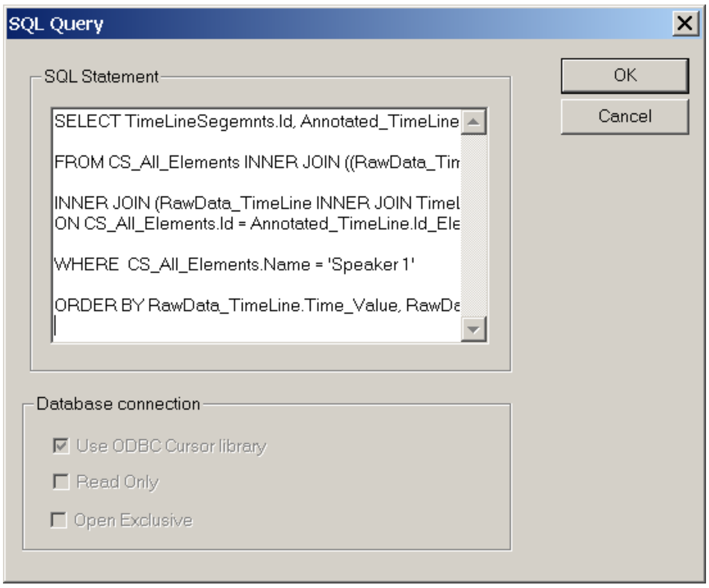

Figure 4.3.1.5: SQL query example.

4.3.2 NXT

The NXT toolkit basically consists of a set of components which a developer may incorporate if he wants when implementing a particular annotation tool. Thus the screen shots shown below are examples of interfaces one can build using the NXT. They do not show parts of a fully functionally tool. The NXT is an open source (apart from the plug-ins developed by DFKI), platform-independent toolkit implemented in Java and using XML for the internal data representation. The plug-ins are a kind of stand-alone programs which also work together with the Anvil annotation tool. They include a Sonogram tool for speech signal graphical representation and analysis and a graphical visual markup tool for inserting markup directly in raw video data.Figure 4.3.2.1 shows an example interface from which a user can view and listen to a video, see a gesture coding palette (to the right) and change a gesture tag in the text display by selecting the desired tag in the coding palette and then clicking on the "Change gesture type" button.

Figure 4.3.2.2 shows the query window which is the interface to the query engine. The "Bookmarks" entry in the toolbar gives an impression of the query language. One can select formula from this entry or enter them freehand and press search (the search button is behind the drop-down menu). Results are shown separately as indicated by the two tabs named "Query" and "Result" in the figure.

Figure 4.3.2.3 shows another interface example. This time there is again a video which one can see and listen to. The annotation window shows word by word transcription in German, syntactical annotation at word level, and a translation into English which is not word to word aligned. The annotation view is analogue.

Figure 4.3.2.4 shows the Sonogram plug-in developed by DFKI. It is a configurable spectrogram application for audio analysis. The main window shows frequencies (linear or logarithmic scale) and the audio signal.

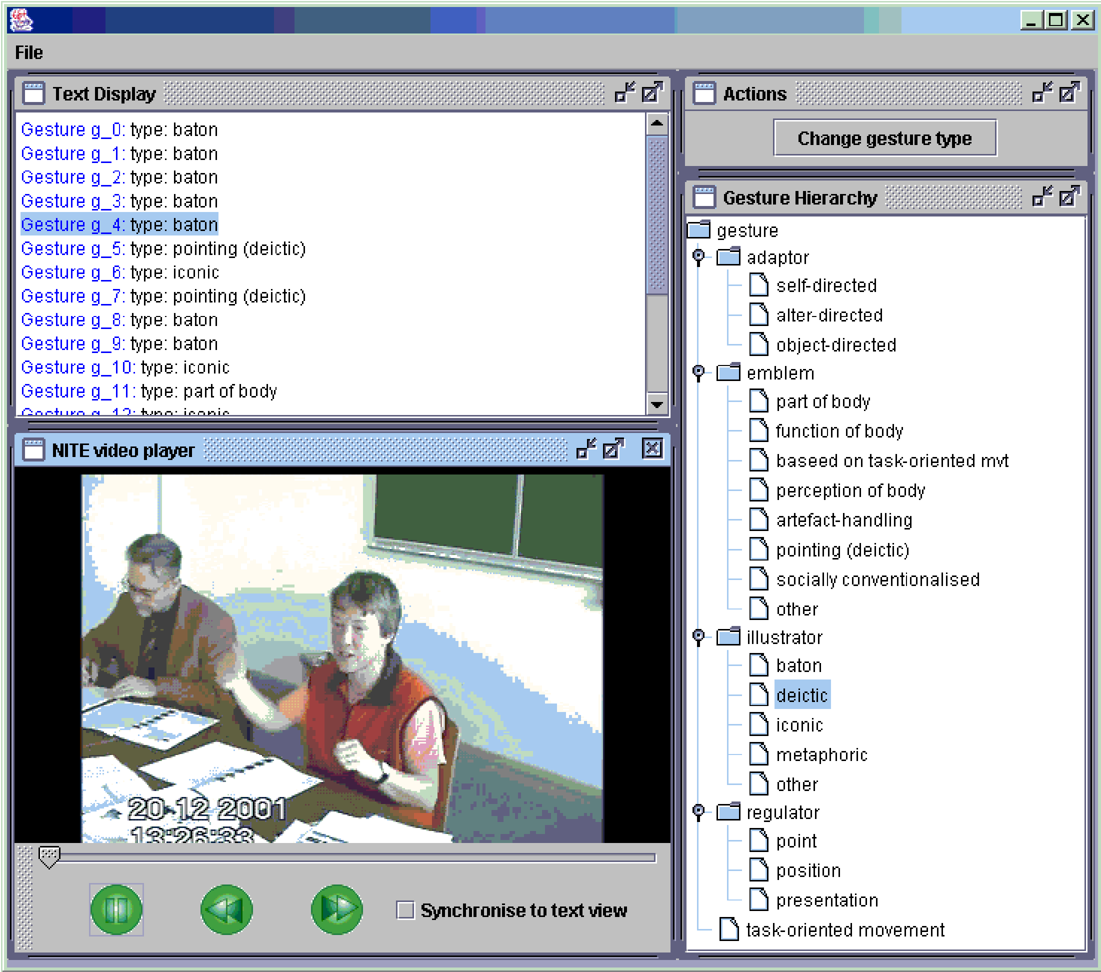

Figure 4.3.2.1: Example interface with gesture coding palette.

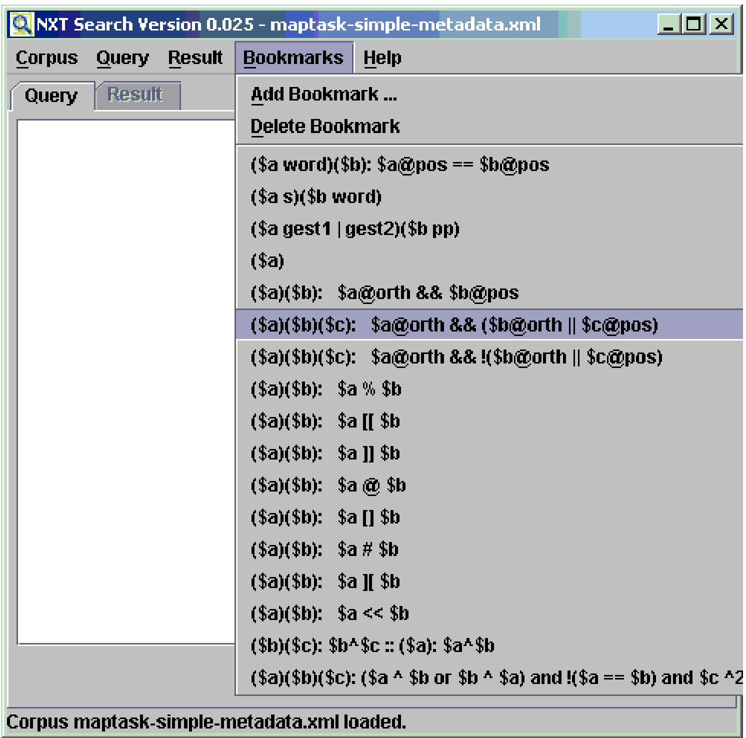

Figure 4.3.2.2: Search window with search formula visible.

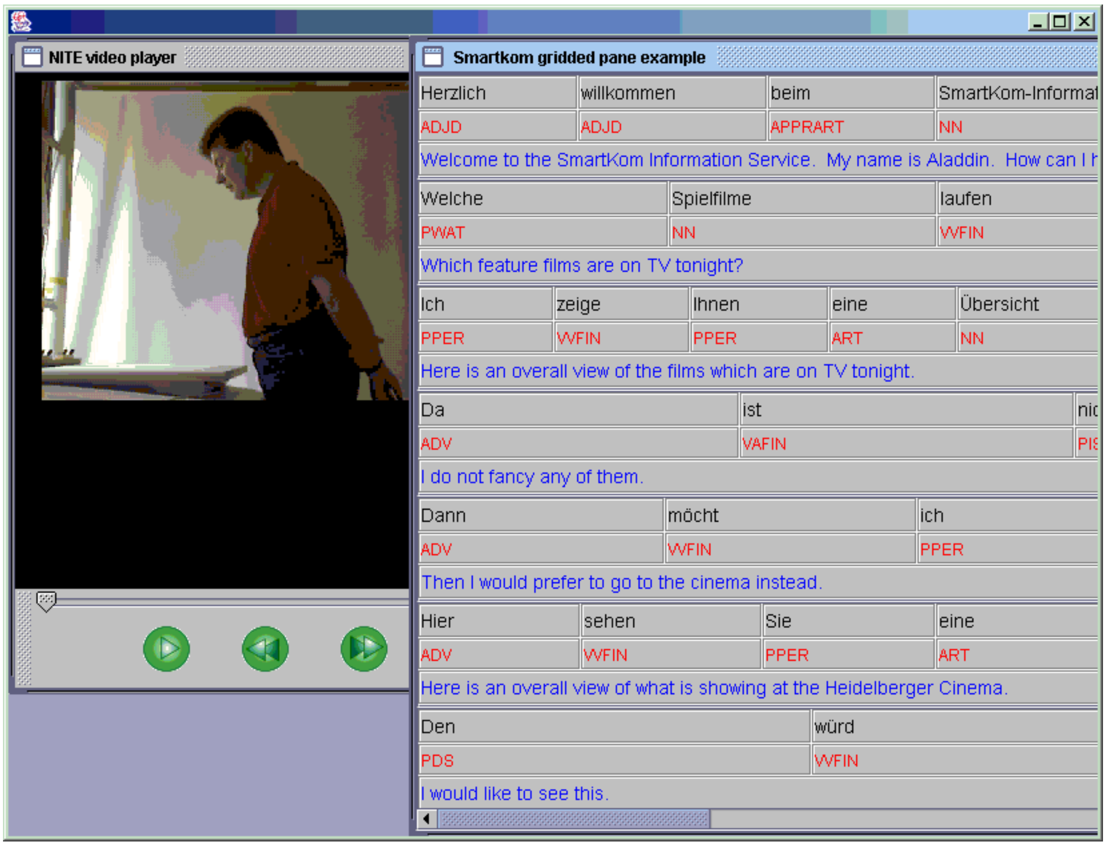

Figure 4.3.2.3: Example interface with analogue coding view.

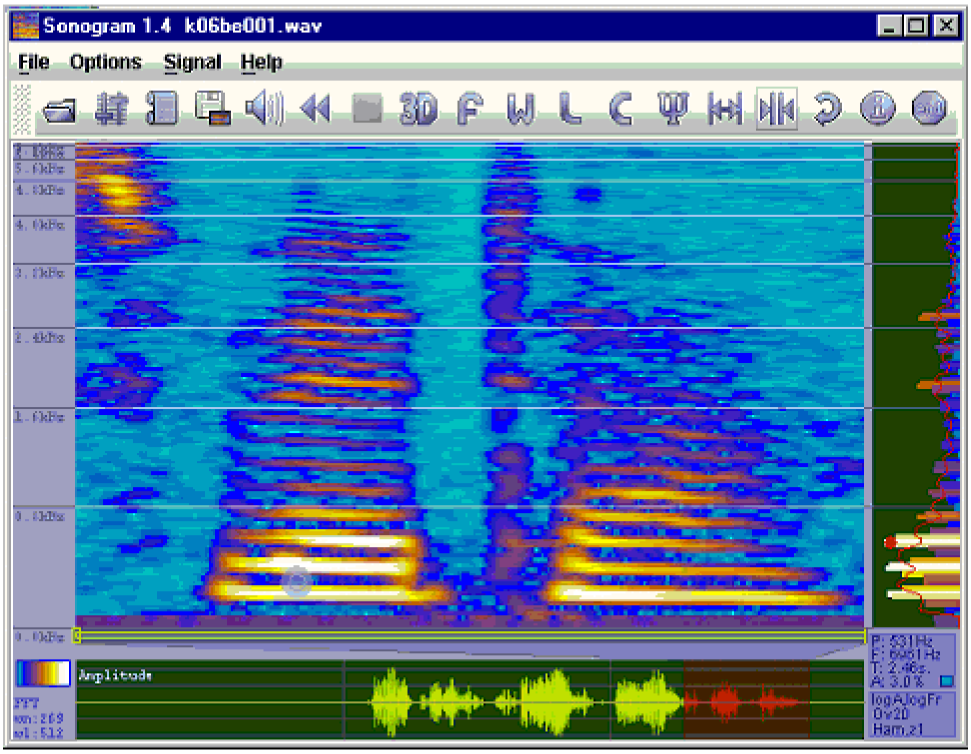

Figure 4.3.2.4: Sonogram plug-in developed jointly with Anvil.

4.3.3 The Observer

The Observer is a commercial tool which has been further developed during the NITE project. The tool is developed in C++. It runs on a Windows platform and uses a database for storage of annotated data. All screen shots shown below are from version 5 which is the most recent commercially available version of the Observer.

Figure 4.3.3.1 shows the opening window in the Observer. To the left one can see the project entitled "a bad day in the office" and one has access to everything which has already been done for this project. Basically the user can do three things with the Observer: (1) He can enter a new configuration. This means entering a new coding scheme and some meta-data. (2) He can do observation. This means that he can annotate a video. (3) He can analyse the coded data.

Figure 4.3.3.2 shows a review of the entered configuration information. The figure only shows part of the review file.

Figure 4.3.3.3 shows how annotation is done in the Observer. It shows a video and a control board for the video. The tag palette is visible to the right at the bottom. Tags are inserted in the table to the left. Tags are time aligned and the bar with the red ends aligns with the video.

Figure 4.3.3.4 show a couple of analysis windows. The Observer allows for a few different types of simple analysis, e.g. intercoder agreement and lag sequential analysis as shown.

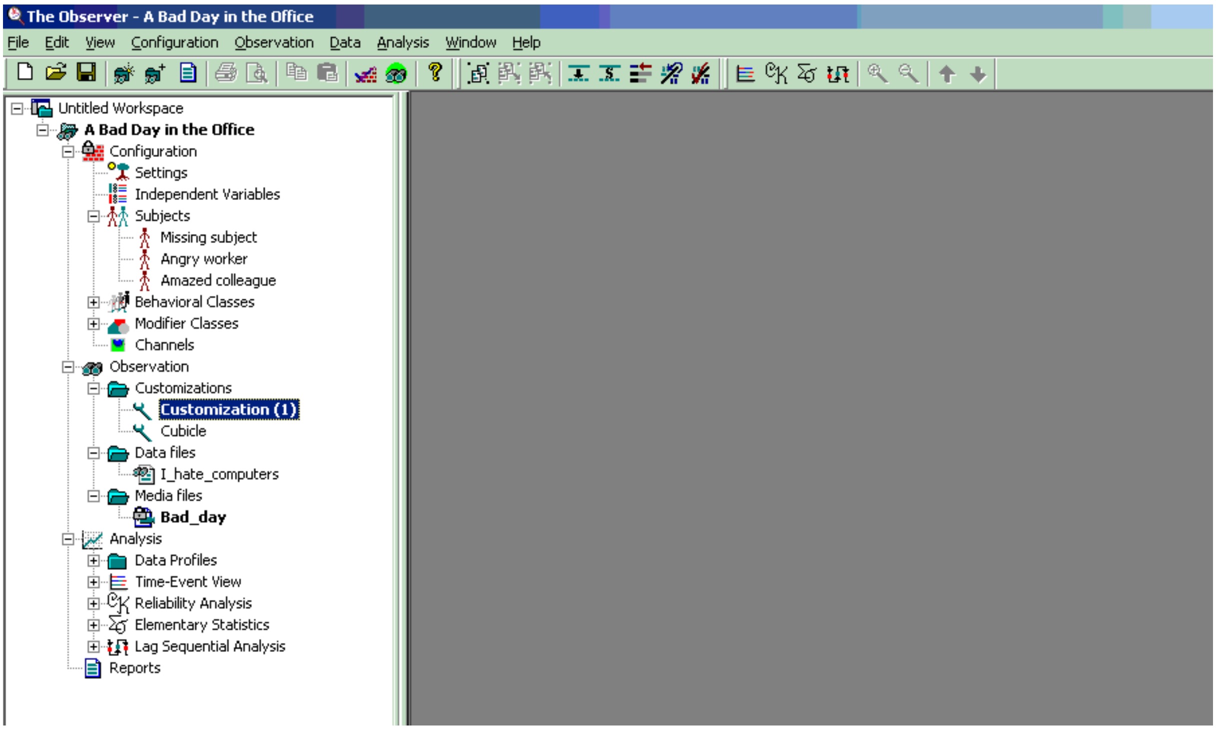

Figure 4.3.3.1: Opening window in the Observer.

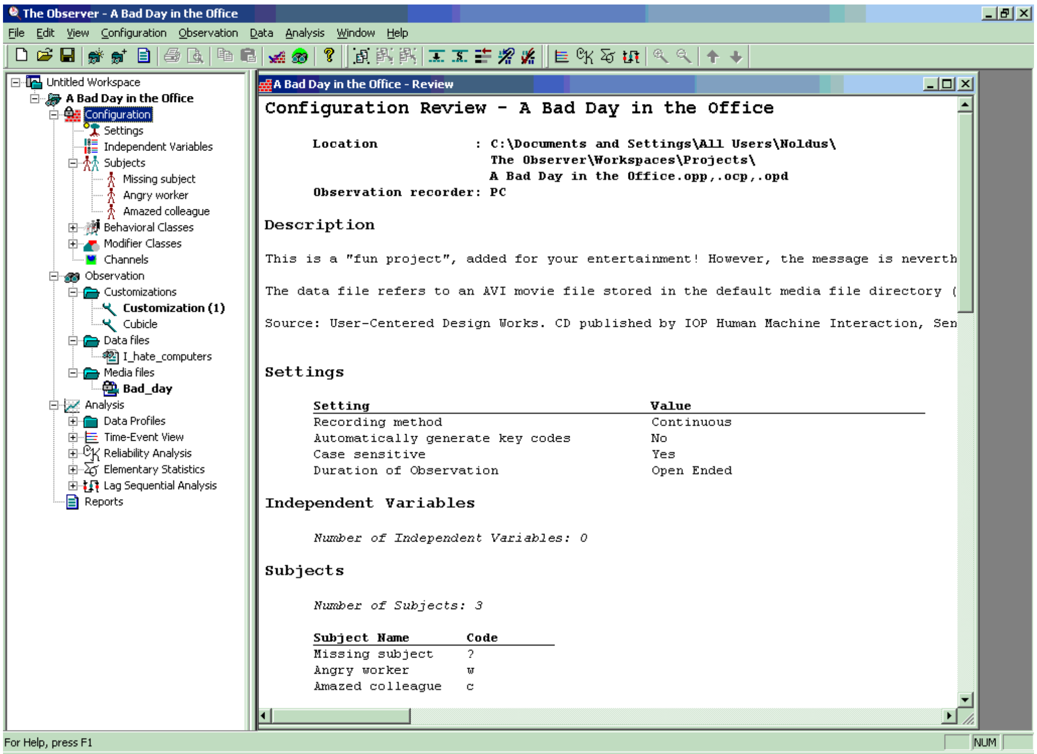

Figure 4.3.3.2: Review of the entered configuration.

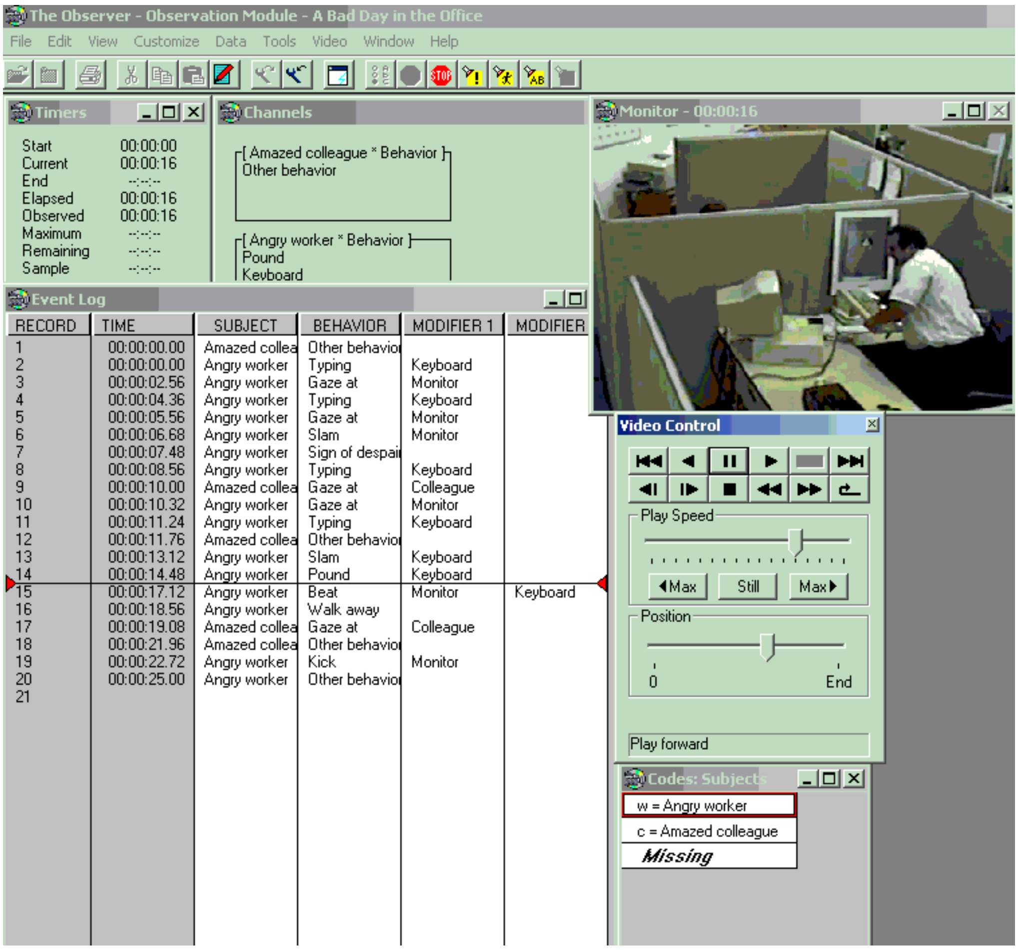

Figure 4.3.3.3: Annotation window in the Observer.

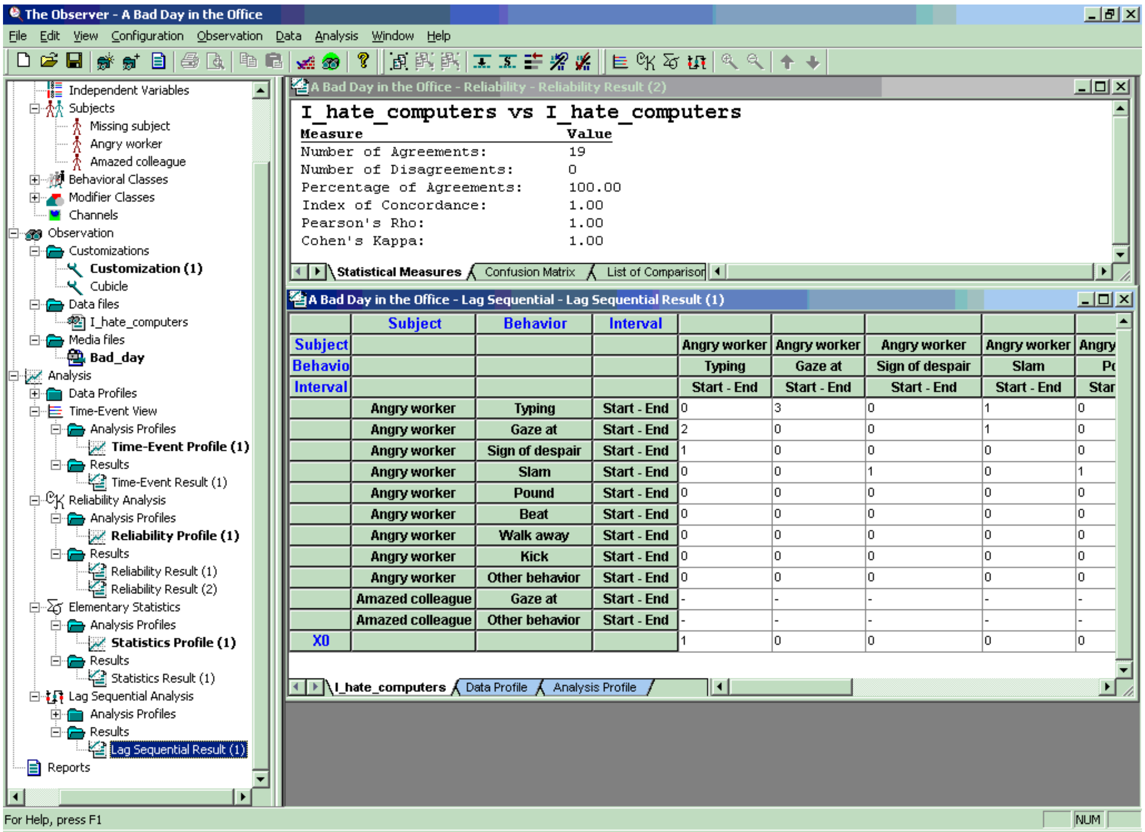

Figure 4.3.3.4: Analysis of coded files with the Observer.

5 Evaluation and assessment

5.1 Validation results

By the start of the NITE project, it was decided that no site would validate its own technical work on the NITE tool strands. Those sites which have had a key role in the implementation of the tools have only been involved in validation to the limited extent of having had the opportunity to comment on validation documents, such as questionnaires, and validation workshop scheduling. The key organisers of tools validation have been the two leaders of the validation workpackage, i.e. ILC and IMS.

With the exception of the DFKI plug-ins, the NITE tools have been evaluated by users from the NITE Advisory Panel (AP) at three occasions, as follows.

The first occasion was the NITE evaluation meeting in Stuttgart in November 2002, documented in D.4.1. The second occasion was the NITE evaluation meeting in Pisa in early June 2003, documented in D.4.2. The third occasion was the remote evaluation orchestrated by NITE partners Pisa and Stuttgart in July-August 2003 in which colleagues at eight NITE AP member sites had the opportunity to use the three NITE tools to code their own NIMM data using their own coding schemes in their home environments. The methodology and results of this third evaluation are also presented in D4.2.

Summarising the findings in the third evaluation round which is probably the more realistic of the three evaluation cycles, partly because of the comparative maturity of the NITE software at the time and partly because of the home-ground test environments, we may observe the following.

The users were asked to evaluate the NITE tools using the following criteria:

1. effectiveness of available functions: does the tool do what it is intended to do?
2. ease of use: how easy is it for an average user to get acquainted with the tool?
3. efficiency of use: does the tool operate in ways users find intuitive and suitable for their purposes?
4. flexibility/customisability/scalability: does the tool operate in ways users can easily modify and adapt to their purposes?

In order to apply the above criteria, users were encouraged to consider the following tools functionalities (cf. Section 4.1 above):

1. creation of a coding project;
2. controlling audio and video;
3. specification of coding schemes;
4. annotation using coding schemes;
5. customised information visualisation;
6. annotation query, retrieval and analysis.

In addition, and just like at the Stuttgart and Pisa evaluation workshops, the NITE AP members were asked for their input on tool functionality which they found missing in the current versions of the tools.

The NITE consortium is extremely grateful for the comments and evaluations received, all of which are listed in D4.2.

It is impossible to draw any solid conclusions concerning the comparative success of the three NITE tool development strands from the in-house tools testings reported in D4.2. Firstly, the tools are so different that comparison in most respects makes little sense, cf. the tool descriptions in Section 4. For instance, The Observer is robust and stable but supports only a modest amount of NIMM coding at present. It has a clear interface design but some users prefer a different interface because they are used to tools which work differently from The Observer. NXT can support a broad range of NIMM coding purposes but only when someone has done the programming needed in each case, and it remains unknown to which extent the tool will scale up to handling large data resources in real time. NWB was, at the time of the trials, the least robust of the three tools but it enables quite complex timestamped coding and is not likely to degrade in performance when handling large data resources.

Secondly, as it turned out, the six functionalities listed above were evaluated by significantly different numbers of users. For the NWB, those six functionalities were evaluated by a total of 22 user/functionality pairs, counting a user/functionality pair each time a particular functionality was tested by a single user (cf. D4.2, Section 5). For The Observer, the number of user/functionality pairs was 10 and each of the six functionalities were evaluated by at least a single user site. For the NXT, the user/functionality pairs number was only 3, meaning that no user site evaluated all six NXT properties and that some (in fact, three) NXT properties were not evaluated by any user site at all. By likely implication, the NWB turned out to receive by far the most comments from the users, in the order of 35+ comments. The Observer and the NXT received far less comments by comparison. There is no explicit evidence that this distribution of NITE tool examinations reflects any preference by the users for an easy-to-use, non-programmers general-purpose NIMM tool but the inference is, of course, a tempting one to make.

Taken together, the NITE user evaluations demonstrate a keen interest in general-purpose NIMM coding tools and significantly contribute to making evident the scale of the task of providing the world's first tool of the kind. We have also learnt that users do not always agree among themselves, conditioned, as they presumably are, by habitually using particular existing special-purpose or limited-purpose tools which, no doubt, incorporate an interaction philosophy different from any of the NITE tools.

5.2 Internal collaboration

The NITE project has brought together researchers with very different skills and backgrounds. As a whole, if may be concluded that the consortium has been up to its ambitious task. Most consortium members knew each other well from, in particular, the MATE project, which may be why the development of the three tool strands have gone rather smoothly. Inevitably, and quite possibly, fortunately, the fact that several software strands were developed in parallel in the project has produced some amount of inter-group competition, in particular between the NWB and NXT teams.

Somewhat surprisingly, especially when comparing to the MATE project, NITE could be claimed to have seen less under-estimation of the time and resources needed for software development than is the case in many other distributed-environment research prototype projects. No fully general-purpose tool has been developed in NITE but, in retrospect, we can safely say that, at the start of the project, none of us really knew what such a tool was and had to be able to do in detail. On the other hand, three tool strands have been developed which are now all being taken up by the research community.

However, the price for increasing project development ambitions may to some extent have been paid elsewhere on the project agenda, i.e. by producing some significant delays in several of the NITE report deliverables. Due to the tall development agendas facing most participants, we may have shifted part of the attention from producing those reports on time to the massive challenges involved in analysing, specifying, and implementing the software strands.

6 Conclusion and future prospects

6.1 Conclusion on the NITE results

NITE has taken a major step towards demonstrating the world's first fully general-purpose coding tool for natural interactivity and multimodal data. In the process, NITE has developed two very different, still limited-purpose according to the definitions in Section 4.2 above, NIMM coding tools which are now being used for NIMM data coding in an increasing number of projects. In addition, the common NITE general-purpose tool specification and the experience on user needs and preferences gathered in the course of the project, are now feeding into the company Noldus's future tool release plans, ensuring an industrial and commercial impact of the work done.

The ISLE report on the state of the art in NIMM coding tools [Dybkjær et al. 2001b] is now more than two years old. It is therefore natural to ask the question: where do the NWB and the NXT stand relative to the current, year 2003 state of the art in limited-purpose NIMM coding tools, there being no general-purpose NIMM coding tools out there as yet? Roughly, recent developments have taken place along the two different lines illustrated by the NWB and the NXT, respectively.

The first line of development has pursued the programmer's toolset idea exemplified by the NXT. Perhaps the primary example of the NXT competition is the US-developed AGTK (Annotation Graph ToolKit) [http://agtk.sourceforge.net/. We know that the AGTK is in use for NIMM data coding at various user sites but have no information about the scale and speed of AGTK dissemination.

The second line has pursued the ready-made tool for ordinary users idea exemplified by the NWB. Perhaps the two primary examples of the NWB competition are (i) DFKI's Anvil tool [http://www.dfki.de/~kipp/anvil] which was developed for gesture markup and which to some extent can assist in spoken dialogue markup in combination with gesture. Like Anvil, (ii) German TASX [http://tasxforce.lili.uni-bielefeld.de/] supports limited-purpose analogue view NIMM coding (see Section 3.1.2) but is perhaps in general slightly more primitive than Anvil. Compared to the NWB, the obvious strength of these tools is that they support the analogue coding view which we want the NWB to support in a future release. By comparison, some weaknesses of those tools are that they incorporate no notion of easy coding scheme entry and that they do not support the symbolic coding view. A more detailed, systematic comparison between the tools mentioned in the present section was presented by NISLab at the Eurospeech tutorial in August 2003.

So, in conclusion, it may be stated that, for general-purpose NIMM data coding, the rapidly growing communities concerned are far better served today than when NITE started. Moreover, two of the most advanced coding tools today have been developed in NITE. On the other hand, as we have tried to make clear in this final report from the NITE project, there is still some way to go before we have an easy-to-use, fully general-purpose coding tool for natural interactivity and multimodal data resources.

6.2 Exploitation planning and perspectives

The NITE partners have individually submitted their exploitation plans, or Technology Integration Plans (TIPs), to the Commission, so we only want to add a single comment here.

Originally, NITE aimed to make the NITE-developed technologies available as open source for colleagues across the world to be able to contribute to their further development. Needless to say, this open source agenda did not apply to Noldus' commercial product, The Observer. When, early on, the consortium decided to iteratively develop two research prototypes, i.e. the

NWB and the NXT, rather than a single “NITE toolset” (see Section 2), we kept the open source agenda. At the time of writing, the NXT is actually open source software, except for the DFKI plug-ins which, as it turned out, DFKI did not want to commit to the open source market. As for the NWB, the current NWB3 release is freeware but not open source because of its inclusion of commercial MS Windows development platform parts. NISLab is still happy to make the in-house-developed NWB software open source. It is just not clear to us if there will be any uptake on our open source offer by developers out there. To find out, we have added an announcement to the NWB release website asking for expressions of interest.

6.3 Future perspectives

The three NITE NIMM coding tools will be maintained in the foreseeable future by NISLab, U. Edinburgh, and Noldus, respectively. All three sites expect to continue tools development and hope that, as a further result of the NITE project, one or more full general-purpose NIMM coding tools will soon be available to users world-wide.

7 References (this document)

Dybkjær, L., Berman, S., Bernsen, N.O., Carletta, J., Heid, U. and Llisterri, J.: Requirements Specification for a Tool in Support of Annotation of Natural Interaction and Multimodal Data. ISLE Deliverable D11.2, July 2001a.

Dybkjær, L., Berman, S., Kipp, M., Olsen, M.W., Pirrelli, V., Reithinger, N. and Soria, C.: Survey of Existing Tools, Standards and User Needs for Annotation of Natural Interaction and Multimodal Data. ISLE Deliverable D11.1, January 2001b.

Dybkjær, L., Bernsen, N.O., Dybkjær, H., McKelvie, D. and Mengel, A.: The MATE Markup Framework. MATE Deliverable D1.2, November 1998.

Knudsen, M. W., Martin, J.-C., Dybkjær, L., Ayuso, M. J. M, N., Bernsen, N. O., Carletta, J., Kita, S., Heid, U., Llisterri, J., Pelachaud, C., Poggi, I., Reithinger, N., van ElsWijk, G. and Wittenburg, P.: Survey of Multimodal Annotation Schemes and Best Practice. ISLE Deliverable D9.1, 2002.

Websites

ISLE: http://www.ilc.cnr.it/EAGLES96/home.html

ISLE NIMM: http://isle.nis.sdu.dk

MATE: http://mate.nis.sdu.dk

NITE: http://nite.nis.sdu.dk

Appendices

8 NITE project deliverables

Del. No	Deliverable name	Del. Type
D1.1	Specification of and plan for workbench architecture, functionality and usability	Specification
D1.1 addendum	List of requirements	Specification
D2.1	Survey of existing gesture, facial expression, and cross-modality coding schemes	Report
D2.2	The NITE markup framework	Report
D2.3	Best practice gesture, facial expression, and cross-modality coding schemes for inclusion in the workbench	Report
D3.1	First NITE workbench software systems	Prototypes
D3.2	NITE workbench software systems release for final validation including full documentation	Prototypes
D3.3	Full documentation of the NITE tools	Report
D4.1	Evaluation report on the first NITE tools	Report
D4.2	Evaluation report on the final NITE tools	Report
D6.2	Annual report	Report
D6.3	Final report and technology implementation plan	Report

9 NITE Publications

Bernsen, N. O., Dybkjær, L. and Kolodnytsky, M.: The NITE workbench. A tool for annotation of natural interactivity and multimodal data. In *Proceedings of the Third International Conference on Language Resources and Evaluation* (LREC’02), Las Palmas, Canary Islands, 2002, 43-49.

Bernsen, N. O., Dybkjær, L. and Kolodnytsky, M.: An Interface for Annotating Natural Interactivity. In J. v. Kuppevelt and R. W. Smith (Eds.): *Current and New Directions in Discourse and Dialogue*, Dordrecht: Kluwer 2003.

Bernsen, N. O., Dybkjær, L. and Kolodnytsky, M.: The NITE Workbench for Annotating Natural Interactive Behaviour. In P. Paggio, K. Jokinen, A. Jönsson (Eds.): *Proceedings of the 1st Nordic Symposium on Multimodal Communication*, Center for Sprogteknologi, Report 6, September 2003, 155-168.

Carletta, J., Kilgour, J., O'Donnell, T., Evert, S., and Voormann, H.: The NITE Object Model Library for Handling Structured Linguistic Annotation on Multimodal Data Sets. Proceedings of the EACL Workshop on Language Technology and the Semantic Web (3rd Workshop on NLP and XML, NLPXML-2003), http://www.ltg.ed.ac.uk/~jeanc/nlpxml2003.final.pdf, 2003, to appear.

Carletta, J., Evert, S., Heid, U., Kilgour, J., Robertson, J., and Voormann, H.: The NITE XML Toolkit: flexible annotation for multi-modal language data. Behavior Research Methods, Instruments, and Computers, special issue on Measuring Behavior, in press.

Carletta, J., McKelvie, D., and Isard A.: Supporting linguistic annotation using XML and stylesheets. In G. Sampson and D. McCarthy (Eds.): *Readings in Corpus Linguistics,* Continuum International, 2002.

Dybkjær, L. and Bernsen, N. O.: Natural interactivity resources. Data, annotation schemes and tools. In *Proceedings of the Third International Conference on Language Resources and Evaluation* (LREC'02), Las Palmas, Canary Islands, 2002, 349-356.

Dybkjær, L. and Bernsen, N. O.: Data Resources and Annotation Schemes for Natural Interactivity: Purposes and Needs. In *Proceedings of the Workshop on Multimodal Resources and Multimodal Systems Evaluation,* held adjacent to the Third International Conference on Language Resources and Evaluation (LREC'02), Las Palmas, Canary Islands, 2002, 1.7.

Dybkjær, L. and Bernsen, N. O.: Data, annotation schemes and coding tools for natural interactivity. In *Proceedings of the International Conference for Spoken Language Processing, ICSLP'2002,* Denver, CO, USA, 217-220, 2002.

Isard, A.: An XML architecture for the HCRC Map Task Corpus. In *Proceedings of the Bidialog Workshop*, University of Bielefeld, Germany, June 2001.

Llisterri, J., Machuca, M.J., de la Mota, C., Riera, M., Ríos, A.: Entonación y tecnologías del habla. In Prieto, P. (Ed.) *Teorías de la entonación*. Barcelona: Ariel (Lingüística), to appear.

Soria, C., Bernsen, N. O., Cadée, N., Carletta, J., Dybkjær, L., Evert, S., Heid, U., Isard, A., Kolodnytsky, M., Lauer, C., Lezius, W., Noldus, L. P. J. J., Pirrelli, V., Reithinger, N. and Vogele, A.: Advanced Tools for the Study of Natural Interactivity. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'2002),* Las Palmas, May 2002.

Two joint publications with the TASX developers (Bielefeld University) are being prepared. They will be published in 2004.

A journal publication and several conference publications on the NWB are being prepared.

10 NITE partners

Natural Interactive Systems Laboratory (NISLab) (coordinating partner)
University of Southern Denmark
Campusvej 55
DK-5230 Odense M, Denmark
Phone: +45 65 50 35 53
Fax: +45 65 50 38 49
Web: http://www.nis.sdu.dk/
Niels Ole Bernsen, contact person (nob@nis.sdu.dk)
Laila Dybkjær (laila@nis.sdu.dk)
Mykola Kolodnytsky (mykola@nis.sdu.dk)

Departament de Filologia Espanyola (DFE)
Universitat Autònoma de Barcelona, Bellaterra,
E-08193 Barcelona, Spain
Phone: +34 93 581 12 16
Fax: +34 93 581 16 86
Joaquim Llisterri, contact person (Joaquim.Llisterri@uab.es)
María Machuca (maria@liceu.uab.es)
Montserrat Riera (montse@liceu.uab.es)

Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI)
Stuhlsatzenhausweg 3
D-66123 Saarbrücken, Germany
Phone: +49 681 302 53 46
Fax: +49 681 302 53 41
Norbert Reithinger, contact person (bert@dfki.de)
Christoph Lauer (clauer@dfki.de)

Human Communication Research Centre (HCRC)
2, Buccleuch Place
Edinburgh EH8 9LW, UK
Phone: +44 131 650 4665
Fax: +44 131 650 4587
Jean Carletta, contact person (j.carletta@edinburgh.ac.uk)
Jonathan Kilgour (jonathan@cogsci.ed.ac.uk)
Tim O'Donnell (timo@cogsci.ed.ac.uk)
Judy Robertson (judyr@cogsci.ed.ac.uk)

Institut für Maschinelle Sprachverarbeitung (IMS)
Universität Stuttgart
Azenbergstraße 12
D-70174 Stuttgart, Germany
Phone: +49 711 121 1373
Fax: +49 711 121 1366
Ulrich Heid, contact person (heid@ims.uni-stuttgart.de)
Stefan Evert (evert@ims.uni-stuttgart.de)
Jan van Kuppevelt (kuppevelt@ims.uni-stuttgart.de)
Holger Voormann (voormann@ims.uni-stuttgart.de)

Istituto Linguistica Computazionale (ILC)
Area della Ricerca di Pisa

via Alfieri 1,
I-56010, S. Cataldo, Pisa, Italy
Phone: +39 050 3152836
Fax: +39 050 3152834
Vito Pirrelli, contact person (vito@ilc.pi.cnr.it)
Claudia Soria (soria@ilc.pi.cnr.it)
Noldus Information Technology b.v.
Costerweg 5
P.O. Box 268
6700 AG Wageningen, The Netherlands
Phone: +31-317-497677
Fax: +31-317-424496
Web: http://www.noldus.com
Niels Cadée, contact person (n.cadee@noldus.nl)
Lucas Noldus, scientific advisor (l.noldus@noldus.nl)
Herman Roosken, manager Research & System Engineering (h.roosken@noldus.nl)
Andrew Spink, manager Research & System Engineering (a.spink@noldus.nl)
Hans Theuws, product manager The Observer (h.theuws@noldus.nl)
Erik Meyer, project leader The Observer (e.meyer@noldus.nl)
Roland Jansen, system analyst (r.jansen@noldus.nl)
Ben Loke, senior system engineer (b.loke@noldus.nl)
Cecilia Herrera, senior system engineer (c.herrera@noldus.nl)
Tijmen van Voorthuijsen, software engineer (t.van.voorthuijsen@noldus.nl)
Leon Wiertz, software engineer (l.wiertz@noldus.nl)
Tiffany Mayton-TheCrow, documentation specialist (t.thecrow@noldus.nl)
Eduard van Middendorp, test engineer (e.van.middendorp@noldus.nl)

Iinstitute for Linguistics, Research Center for Semiotics (subcontractor to IMS, Stuttgart)
Technical University of Berlin
Ernst-Reuter-Platz 7
10587 Berlin, Germany
Phone: + 49 30 3142 3115
Fax: + 49 30 3142 7638
Massimo Serenari, contact person (serenari@cs.tu-berlin.de)

11 NITE Advisory Panel Members

Australia

Robert Pedlow
Telstra Coporation
Telstra Centre for Accessibility
Clayton, Vic 3168
robert.pedlow@team.telstra.com

France

Jean-Claude Martin
LIMSI-CNRS/LINC-IUT de Montreuil
Paris
martin@limsi.fr

Laurent Romary
Loria-CNRS
Langue et Dialogue
Vandoeuvre Les Nancy
Laurent.Romary@loria.fr

Germany

Melina Alexa
Brockhaus Duden Neue Medien GmbH
Mannheim
alexa@bifab.de

Gregor Erbach
Saarland University
Computational Lingustics Department
Saarbrücken
gor@acm.org

Detlef Ruschin
Heinrich-Hertz-Institut fuer
Nachrichtentechnik Berlin GmbH
Dep. Interactive Media - Human Factors
Berlin
Ruschin@HHI.DE

Thomas Schmidt
University of Hamburg
SFB "Mehrsprachigkeit" (Research Center on Multilingualism)
Hamburg
thomas.schmidt@uni-hamburg.de

Michael Strube
European Media Laboratory GmbH
Heidelberg
Michael.Strube@eml.villa-bosch.de

Italy

Michele Cornacchia
Fondazione Ugo Bordoni
Human Factors Group
Rome
mcornacchia@fub.it

Morena Danieli
LOQUENDO
Torino
morena.danieli@loquendo.com

Giacomo Ferrari
Università del Piemonte Orientale
Dipartimento di studi Umanistici
Vercelli
ferrari@lett.unipmn.it

Japan

Masato Ishizaki
Japan Advanced Institute of Science and Technology
Ishikawa
masato@jaist.ac.jp

Netherlands

Jan Peter de Ruiter
Max Planck Institute for Psycholinguistics
Nijmegen
janpeter.deruiter@mpi.nl

Spain

Daniel Tapias
Telefónica Móviles España
Voice products and services development Dept.
Madrid
tapias_d@tsm.es

Sweden

Jens Allwood
Göteborg University
Department of Linguistics
Göteborg
jens@ling.gu.se

Nils Dahlbäck
Linköping university
NLPLAB
Linköping
nilda@ida.liu.se

Switzerland

Hervé Bourlard
Swiss Federal Institute of Technology at Lausanne (EPFL)
IDIAP
Martigny
bourlard@idiap.ch

United Kingdom

Eric Atwell
University of Leeds
Distributed Multimedia Systems
Leeds
eric@comp.leeds.ac.uk

Mark Huckvale
University College London
Phonetics and Linguistics
London
mark@phon.ucl.ac.uk

Stephen J. McKenna
University of Dundee
Department of Applied Computing
Dundee
stephen@computing.dundee.ac.uk

Johanna D. Moore
University of Edinburgh
Human Communication Research Centre
Edinburgh
J.Moore@ed.ac.uk

Massimo Poesio
University of Essex
poesio@essex.ac.uk

Steve Renals
Sheffield University
Dept of Computer Science
Sheffield
s.renals@dcs.shef.ac.uk

Amanda Schiffrin
The University of Leeds
AI Laboratory
Leeds
mandy@comp.leeds.ac.uk

Chris Taylor
Universtiy of Manchester
Imaging Science and Biomedical Engineering
Manchester
Chris.Taylor@man.ac.uk

Bonnie Webber
University of Edinburgh
Edinburgh
bonnie.webber@ed.ac.uk

USA

John Aberdeen
MITRE
Bedford, MA
aberdeen@mitre.org

Steven Bird
University of Pennsylvania
CIS & Linguistics
Philadelphia, PA
Steven.Bird@ldc.upenn.edu

Jane A. Edwards
UC Berkeley
Institute of Cognitive and Brain Sciences
edwards@icsi.berkeley.edu

Lisa D. Harper
The MITRE Corporation
McLean, VA
lisah@mitre.org

Diane Litman
University of Pittsburgh
litman@cs.pitt.edu

Brian MacWhinney
Carnegie Mellon University
Department of Psychology
Pittsburgh, PA
macw@cmu.edu

David McNeill
University of Chicago
dmcneill@uchicago.edu

David R. Traum
USC/ICT
Marina del Rey, CA
traum@ict.usc.edu

Marilyn Walker
AT&T Labs Research
Florham Park, NJ
walker@research.att.com