## References


1. S. Attardo, *Linguistic Theory of Humor*, Mouton de Gruyter, 1994.
2. K. Binsted and G. Ritchie, "Computational Rules for Punning Riddles," *Humor: Int'l J. Humor Research*, vol. 10, no. 1, 1997, pp. 25–76.
3. O. Stock and C. Strapparava, "Getting Serious about the Development of Computational Humor," *Proc. 18th Int'l Joint Conf. Artificial Intelligence* (IJCAI 03), Morgan Kaufmann, 2003, pp. 59–64.
4. R. Mihalcea and C. Strapparava, "Laughter Abounds in the Mouths of Computers: Investigations in Automatic Humor Recognition," *Proc. Intelligent Technologies for Interactive Entertainment* (INTETAIN 05), LNCS 3814, Springer, 2005, pp. 84–93.
5. C. Bucaria, "Lexical and Syntactic Ambiguity as a Source of Humor," *Humor: Int'l J. Humor Research*, vol. 17, no. 3, 2004, pp. 279–309.
6. A. Valitutti, C. Strapparava, and O. Stock, "Lexical Resources and Semantic Similarity for Affective Evaluative Expressions Generation," *Proc. 1st Int'l Conf. Affective Computing and Intelligent Interaction* (ACII 05), LNCS 3784, Springer, 2005, pp. 474–481.


## The STANDUP Interactive Riddle-Builder


Graeme Ritchie, *University of Aberdeen*
Ruli Manurung and Helen Pain, *University of Edinburgh*
Annalu Waller and Dave O'Mara, *University of Dundee*


As children grow up, their language and communication skills develop as a result of their experience in the world and their interactions with other language users. In many cultures, an important type of interaction is language play with the child's peer group—word games and joke telling. A child with a disability such as cerebral palsy might converse using a voice output communication aid—a speech synthesizer attached to a physical input device. This cumbersome way of talking tends to isolate a child from the repartee, banter, and joke telling typical of the playground. This lack of practice can inhibit the development of language skills, leading to a lack of conversational fluency or even an undermining of social skills.

Our STANDUP (System to Augment Non-speakers' Dialogue Using Puns) project aims to take a small step toward alleviating this problem by providing, in software, a language playground for children with disabilities. The program provides an interactive user interface, specially designed for children with limited motor skills, through which a child can create simple jokes (riddles based on puns) by selecting words or topics. Here are two typical jokes that the system produces:

- What kind of berry is a stream? A current currant.
- How is an unmannered visitor different from a beneficial respite? One is a rude guest, the other is a good rest.

The system isn't just an online jokebook. It builds new jokes on the spot using 10 simple patterns for the essential shapes of punning riddles and a lexical database of about 130,000 words and phrases. We hope children will enjoy using the software to experiment with sounds and meanings to the benefit of their linguistic skills.


The program provides an interactive user interface, specially designed for children with limited motor skills, through which a child can create simple jokes by selecting words or topics.


### Background

Although various researchers have attempted since 1992 to get computers to produce novel jokes,[1] STANDUP's main predecessor is the JAPE system.[2] That program could churn out hundreds of punning riddles, some of which children judged to be of reasonable quality. However, it was only a rough research prototype; it took a long time to produce results and had no real user interface, and the ordinary user couldn't control it. We've used JAPE's central ideas to create a fully engineered, large-scale, interactive riddle generator with a user interface specially aimed at children with disabilities.

### Designing with users

As computational humor is still in its infancy, we had no precedent for real-world use of a system such as STANDUP and certainly no experience of providing a joke generator for children with disabilities. We therefore devoted a substantial portion of the project to user-centered design. We consulted potential users and associated experts (teachers and speech and language therapists) about how the system, particularly the user interface, should operate.[3,4] In the early stages, we used nonsoftware mock-ups. We showed users laminated sheets representing screen configurations and asked them to step through tasks by pointing to the buttons in the pictures. The experimenter responded by replacing each sheet with the appropriate next screen shot. We adopted this low-tech approach to emphasize to the participants that the system hadn't yet been built and that suggestions or criticisms at this stage could genuinely influence the software's eventual design. Experience had shown that software mock-ups, particularly if very slick, give the impression that a working program is already available. This discourages participants from asking for changes and can even distract them into asking how they can get hold of this apparently working program.

On the basis of our studies' results, we built a software mock-up of the user interface (with a dummy joke generator) and tested it for usability with suitable users. Teachers and therapists again gave their advice.

In parallel with this, we designed and implemented the joke-generating back end. This incorporated a wide variety of facilities for manipulating words and joke structures, but our studies with users and experts suggested that this particular user group needed only a subset of these.

### How the system works

You can view the STANDUP program as having two relatively separate major parts: the front end, which embodies the user interface and controls any user options, and the back end, which manages the lexical database and generates the jokes.

The user interface displays three main areas: the general navigation bar, the joke-selection menu, and the progress chart (see figure 3). The navigation bar is a standard set of buttons for going back, forward, exiting, and so on. The joke-selection menu consists of large labeled buttons through

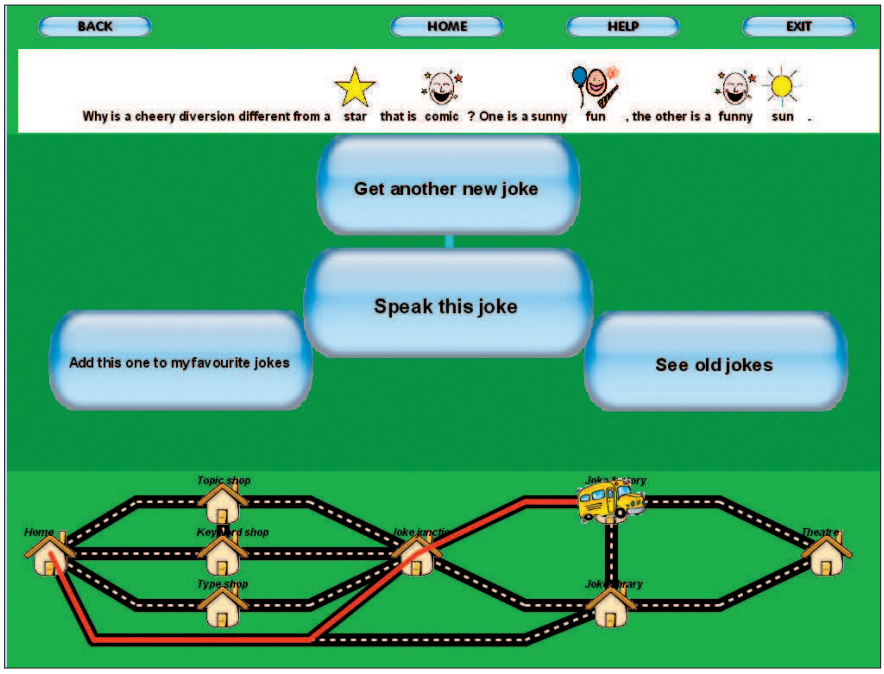



**Figure 3. The STANDUP user interface.**

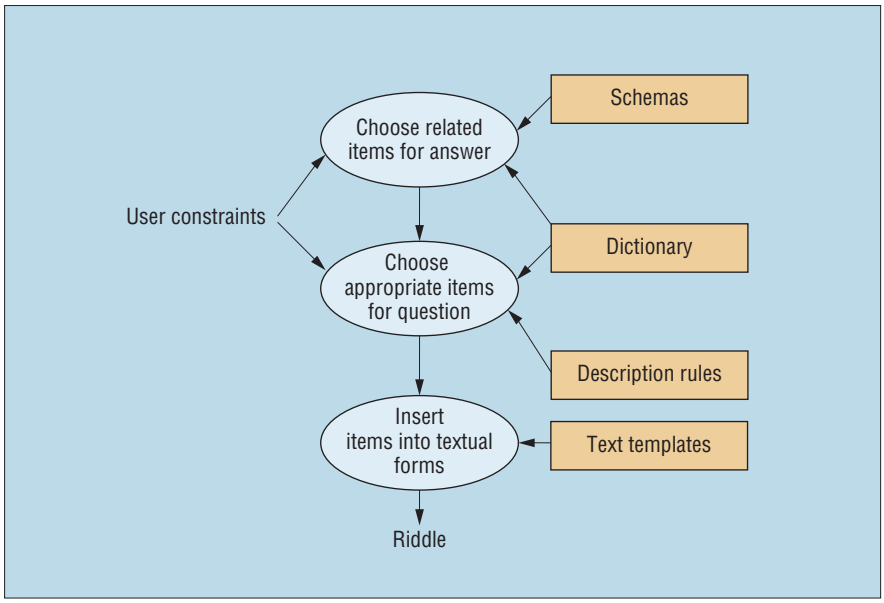



**Figure 4. The structure of the STANDUP back end.**

which the user controls the joke generator using a standard mouse, a touch screen, or a single-switch scanning interface (routinely adopted for those with limited motor skills). The progress chart shows where the user is in creating or finding a joke, using the metaphor of a bus journey along a simple road network, with stops such as "Word Shop" and "Joke Factory."

The back end (see figure 4) contains several components: a set of schemas, a set of description rules, a set of text templates, and a dictionary. The schemas define the linguistic patterns underlying punning riddles. For example, a schema might represent information such as

*Find items X, Y in the dictionary such that X is a noun or adjective, Y is a noun, and X and Y sound the same.*

This describes the central relationships in an example like the current/currant joke shown earlier.

The schema contains information that mainly constrains the ingredients of the riddle's answer, because that's where the pun occurs in all the types of riddle we've used. Once the system finds suitable items to match a schema, it passes these dictionary entries on to the description rules, which flesh out the descriptive phrases needed for the question. For example, starting from "rude" and "guest," it might try to find a way to build a phrase describing or meaning the same as "rude guest," such as "unmannered visitor," or it might select two items that can be "crossed" to produce "rude guest," such as "boor" and "visitor."

For the third stage, text templates contain canned strings such as "What do you get when you cross a ... and a …" or "What do you call a ..." alongside labeled slots into which the template-handling software slots the words and phrases provided by the schemas and the description rules, thus producing the final text.

The user can control this process via the graphical interface by imposing constraints on answer building (the schema), question building (the description rules), or both. For example, the user might specify that the joke must contain a particular word, be on a particular topic, or be a particular type of riddle.

Joke creation depends heavily on the dictionary, which contains information from numerous sources in a relational database. We took syntactic categories (such as noun and verb) and semantic relations (synonymy, being a subclass of, and so on) from the public-domain lexicon WordNet,[5] which contains approximately 200,000 entries. For information about the sound of words, we used the Unisyn pronunciation dictionary to turn a word or phrase's ordinary spelling into a standard phonetic notation. For our final user trials, we attached pictures to as many of the words as possible, using proprietary symbol sets that two companies involved in creating software aids for disabled users lent to us. The resulting lexical database of around 130,000 entries contains several tables, each representing one important relationship (such as between a word and its pronunciation or a word and its synonyms). In this way, we were able to implement dictionary searches as SQL database queries.

Although the joke-generation ideas are relatively simple and have already been tested in principle in the JAPE project, designing and implementing a large-scale, efficient, robust, easily usable system involved considerable work. We tried to make our designs as general as possible and to automate as much of the dictionary creation as possible, so that it should be relatively easy to create revised versions of the dictionary (for example, from a new version of WordNet). We also hope to make the lexical database available to other projects.

### How will children use it?

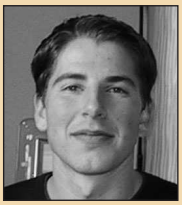
We're about to start our final evaluations of the full system with users. This will involve visiting schools in the surrounding area, both special-needs establishments and mainstream schools. There we shall see how children—both with and without language-impairing disabilities—use the system. Beforehand, we'll assess certain aspects of each child's literacy to give us a context for interpreting what we observe. The time available to us (a few months) isn't sufficient for a long-term study of the software's effects. However, we'll carry out some tests at the end of a child's exploration of the system to see if the sessions have been beneficial in any way.

This project is very much an exploration of possibilities, and we don't know what we'll find out. However, we hope that the work will help move computational humor from tentative research to practical applications. In particular, a software language playground like this could well be of wider use in educational settings. ■

## Acknowledgments

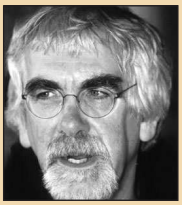
Grants GR/S15402/01 and GR/R83217/01 from the UK's Engineering and Physical Sciences Research Council supported this work. We're grateful to Widgit Software and Mayer-Johnson LLC for their help.

**Kim Binsted** is an assistant professor in Information and Computer Sciences at the University of Hawaii. Contact her at binsted@hawaii.edu.



**Benjamin Bergen** is an assistant professor in the Linguistics Department at the University of Hawaii, Manoa, where he heads the Language and Cognition Laboratory. Contact him at bergen@hawaii.edu.

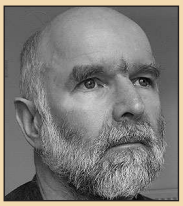
**Seana Coulson** is an associate professor in the Cognitive Science Department at the University of California, San Diego, where she heads the Brain and Cognition Laboratory. Contact her at coulson@cogsci.ucsd.edu.



**Anton Nijholt** is a professor and chair of human-media interaction in the University of Twente's Department of Computer Science. Contact him at anijholt@cs.utwente.nl.

**Oliviero Stock** is a senior fellow at and former director of ITC-irst (Center for Scientific and Technological Research). Contact him at stock@itc.it.

**Carlo Strapparava** is a senior researcher at ITC-irst in the Communication and Technologies Division. Contact him at strappa@itc.it.



**Graeme Ritchie** is a senior research fellow in the University of Aberdeen's Department of Computing Science. Contact him at gritchie@csd.abdn.ac.uk.

**Ruli Manurung** is a research fellow in the University of Edinburgh's School of Informatics. Contact him at ruli.manurung@ed.ac.uk.

**Helen Pain** is a senior lecturer in the University of Edinburgh's School of Informatics. Contact her at h.pain@ed.ac.uk.

**Annalu Waller** is a lecturer in the University of Dundee's Division of Applied Computing. Contact her at awaller@computing.dundee.ac.uk.

**Dave O'Mara** is a postdoctoral research assistant in the University of Dundee's Division of Applied Computing. Contact him at domara@computing.dundee.ac.uk.

## References

1. G. Ritchie, *The Linguistic Analysis of Jokes*, Routledge, 2004.
2. K. Binsted, H. Pain, and G. Ritchie, "Children's Evaluation of Computer-Generated Punning Riddles," *Pragmatics and Cognition*, vol. 5, no. 2, 1997, pp. 309–358.
3. R. Manurung et al., "Facilitating User Feedback in the Design of a Novel Joke Generation System for People with Severe Communication Impairment," *Proc. 11th Int'l Conf. Human-Computer Interaction* (HCI 05), CD-ROM, Lawrence Erlbaum, 2005.
4. D. O'Mara et al., "The Role of Assisted Communicators as Domain Experts in Early Software Design," *Proc. 11th Biennial Conf. Int'l Soc. for Augmentative and Alternative Communication*, CD-ROM, ISAAC, 2004.
5. C. Fellbaum, *WordNet: An Electronic Lexical Database*, MIT Press, 1998.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/publications/dlib.