5SSEL026 – Language Construction
Lecture 8
Artificial Languages 2

Homies an Palones! (an homipalones.) Bona ta vaada yer dolly old ekes agin. I ope yerv aw got yer bijou buns letted an yer feelin fantabulosa. Now, jooj up yer riah, slap on yer oglefakes an start exercisin yer luppers creevin. It's showtime!
This introduction is written in Polari, a Gay cant used from the late 1880s through to the 1980s. It has now largely died out, but a few experienced users still remain.

WHY MAKE ARTIFICIAL LANGUAGES (2)?
Last week we looked at some of the reasons people make artificial languages. This week we will look at some more reasons:

- **To represent nonhumans in fiction.** Many sci-fi and fantasy books make use of the occasional fragment of unrecognisable language to indicate difference; but for some authors this hint of otherness is not enough. For these people, the language, culture and history of their creations need to be set out in some detail if they are to be properly represented to the reader. Perhaps the most extreme case was J. R. R. Tolkien. However, for most authors, their fictional languages are only properly codified when they are converted for a visual medium (film or TV). You may decide to flesh out one of these underdefined languages for your language project.
- **To keep secrets in plain sight.** All languages keep secrets, they exclude anyone who does not know the language from accessing the information being exchanged. However, they are also quite leaky: anyone can learn the language and overcome the barrier. Secret languages, technical languages, cants, groupspeak, even academic register, are all subsets of a language where an extra level of secrecy has been added; they are restricted codes which both limit access to the information being shared and allow speakers to more easily differentiate in-group from out-group. Polari (as used in the introduction, above) is a particularly developed example of a cant.
- **Because not all languages are for exchanges between humans.** While computer languages are the most recent example of this, they come from a long tradition of signalling to domestic animals. For instance, shepherds have complex whistling systems for working with their dogs. However, the development of computer languages over the past seventy years offers an informative compressed timescale for the development of communication between humans and nonhumans; not only have computer languages become more sophisticated in the information they can convey, computers themselves have undergone a steady and relentless increase in their capacity to understand us. How this has happened provides an analogy for the origins of language itself.
- **For fun!** This is perhaps the main reason for creating artificial languages. We are sometimes labelled as Homo ludens[1] (the game-playing human), so why not play games with language?

FANTASY LANGUAGES
Fictional languages are often invented by an author because the fantasy world they are creating has become quite real for them. They feel the need to define not only contexts for their characters and events, but also co-texts and sub-texts. It is not enough to know that a character is a member of tribe X, and tribe X does Y, it is necessary to show how the tribe came to do what they do. A large part of the co-text is the language they use to do Y, and the language has to accommodate the sub-texts of why Y is done.

Fictional languages often start as just a collection of sounds (or script forms) which indicate that something non-English is happening. However, they can then build into word-forms which represent meanings which map quite closely back to English. Some go on to become syntactic systems which often follow the syntax of English; and a few develop grammatical systems which finally begin to distance them from their English origins. It's a bit like the birth of a new natural language, but in a considerably compressed timescale. Often the original author ceases to be interested in further development, and either enthusiasts take it forward (e.g. Klingon) or transfers to film or TV demand more realism (e.g. Dothraki). Occasionally the original author does the whole job himself (e.g. J. R. R. Tolkien's Quenya Elvish).

Sometimes the author (or someone else) adds a specialist script. This is the case with Klingon and, to a certain extent, Tolkien's languages. In the actual world, writing systems are culturally bound to the languages they encode; but in the fantasy world they often represent prejudices about what counts as writing – they are culturally bound to the writing system of the language inventor's first language. Writing – making a permanent record of spoken language – has been invented at least five times in human history: Egyptian (pictograms), Sumerian (ideograms), Harappan (syllabograms), Chinese (logograms), and Olmec (pictograms). There have also been some novel interpretations of pre-existing systems (e.g. Korean, Cyrillic, Hiragana/Katakana, Arabic, Cherokee) which have introduced new scripting concepts; and there is at least one inexplicable outlier (Inca Quipu strings) which we still don't understand.

HOWEVER, YOU SHOULD NOT CREATE A SCRIPT FOR THIS MODULE ASSIGNMENT! Stick to the Roman alphabet (and Roman sounds), supplemented by accent markers if you wish. Alternatively, you can use a phonetic alphabet, or mix phonetic symbols with the Roman alphabet.

Famous fictional language creators include:

J. R. R. Tolkien, writer of *The Hobbit*, *The Lord of the Rings*, and *The Silmarillion*. Tolkien created a detailed fantasy world with a long history, a range of cultures each with their own traditions, legends and heroes, and many languages. He began work on his Elven series of languages in 1910, and continued until his death in 1973. By then he had substantial languages for Quenya and Sindarin, sufficient languages for Eldarin and Avarin, and several other sketched-out Elven languages[2]. He also produced the sufficient languages of Khuzdul (Dwarvish), Númenórean, Adûnaic, Valarin and Black Speech, and sketched out many others.

Marc Okrand, designer of languages for the *Star Trek* franchise. Tolkien produced languages for fun; Okrand was the first to be employed to create a fantasy language. Initially, he created sufficient Vulcan for the ST2 movie, sufficient Klingon for ST3, 5 and 6, and sufficient Romulan for ST2009. Klingon has been adopted by a large fan base, and now has a few thousand users, but probably only 25 fluent speakers. Nonetheless, this is sufficient to make it the most popular fantasy language: it has its own language institute and can now be classed as a substantial language. In 2017 the US courts ruled that Paramount could not hold a copyright over the language or the word Klingon because it had "escaped" into public domain.

Okrand also designed the Atlantean language for the Disney film *Atlantis: The Lost Empire*.

David J. Peterson, designer of languages for *Game of Thrones* and *Defiance*. Peterson won a Language Creation Society competition to design the GoT languages Dothraki and Valyrian, and was later commissioned to produce the Castithan, Irathient and Omec languages for Defiance. Since 2014 he has been involved in a series of projects, creating a further thirteen languages for TV and cinema.

Dothraki is a sufficient language designed around horses, and there are many equine references in it. Valyrian, a substantial language, also has many interesting features (e.g. four genders: lunar, solar, terrestrial, or aquatic), but no script was designed for it. The *Defiance* languages do have scripts, and Castithan, a sufficient language, has a script which is syllabic rather than alphabetic. Each new language that Peterson creates seems to be more complex than the previous one, but it seems that Valyrian remains the language he is most committed to.

Paul R. Frommer, designer of Na'vi for *Avatar* and Barsoomian for *John Carter of Mars*. Frommer was contracted by James Cameron, writer of *Avatar*, to flesh out his ideas for a Na'vi language. Cameron has already decided on about 30 words, so Frommer had to work from a pre-set phonological base to generate a language sufficient for the film's requirements. He introduced some unusual grammatical features to the language (e.g. tense marking in the middle of verbs) and established a distinct phonology, always remembering that the final language had to work within the speech ranges of human actors. By the time the film was finished, Na'vi had a vocabulary of over 1,000 words, and was well within the range of a sufficient language. The film established a fan base for the language, who have since developed Na'vi into a substantial conlang. It even has an official script (and several unofficial ones), despite Na'vi being identified in the film as a language without writing.

Frommer went on to develop Barsoomian for *John Carter of Mars*. He settled on a simple letter replacement code, but the language is multimodal: lexical words (nouns and verbs) are spoken, but grammatical words (pronouns, prepositions, connectors, articles) are transmitted telepathically. This makes speech quite staccato. Once again, Frommer was constrained by the original description of the language; but Edgar Rice Burroughs had been more fanciful in his linguistic ideas than James Cameron, creating a language that was just too alien. Barsoomian has, therefore, neither the completeness nor the fan following of Na'vi.

SECRET LANGUAGES

Secret languages develop because a group of people need to hide ideas in plain sight. They need to be able to exchange information in the presence of others so that those others cannot access the exchange. Essentially, therefore, a secret language is a code; however, unlike the codes used by security services worldwide, they are easily accessible by anyone who joins the encoding community and is willing to make the effort to learn the code. It is possible to become fluent in a secret language.

Secret languages are often not fully separated from local languages and use a lot of their systems – lexis and grammar particularly. For instance:

- **Pig latin** is a simple phonic reordering and incrementing of syllables in otherwise standard English (remove the first consonant cluster from a word and place it at the end, followed by "ay" – igpay atinlay). It relies on the fact that we do not immediately interpret the sounds as English, and we need to practice both speaking and listening to produce and understand it. Once we are "tuned in", we have no problems. The secrecy is in the unusualness of the form, not any complex systematics. To do the written equivalent of pig-latin, you can use another writing system (e.g. the futhark or ogham script or Cyrillic, or any fantasy script that uses simple letter replacement).
- **Polari** relies on English as a base structure but uses a variant lexis (mostly nouns, verbs and adjectives). English grammar and "little words" are largely unaffected. A listener needs to know the new words to understand the code, but that is just learning new vocabulary; and that is something we do when we enter any new community of interest, such as a scientific discipline. Polari's origins involve a mixture of codes and languages, not all of which are English based. It has roots in several predecessor cants, including Elizabethan thieves' cant, 18th Century Molly-house cant, 19th century prostitute cant, 19th century theatrical language, Cockney, Yiddish, and Romani, to name a few.
- **Twinspeak** (also known as idioglossia or cryptophasia) is perhaps the most secret form of secret language. It is a special code developed between close siblings, especially twins; and It often starts as just new lexis – shared special words – but it can later develop new grammar and phonology. Often, because the twins share a great deal of context, the signs are much-reduced markers of quite complex meanings. The most notorious example of twinspeak was June and Jennifer Gibbons, and it didn't end happily. After 14 years in Broadmoor for the crimes of theft, arson and being seriously weird, they decided that one of them could live a normal life if the other died. Soon after, Jennifer died of a heart attack – no foul play was discovered. June has lived a normal life ever since. The Gibbons case was, however, an extreme case; most twinspeak is much more benign, and it disappears when the twins go to school and develop their own personalities.

WORKING WITH MACHINES

Computers were a product of code-breaking in WWII. The German military put a lot of faith in a coding machine called Enigma, which was supposed to produce almost uncrackable ciphers. The code was changed daily, so even if the code for one day was cracked, it did not help with the following day's messages. However, the Allies had Alan Turing, who made another machine which could break the daily code within minutes. It relied on a daily weather report (which always started with the words, "the 6am weather report" and ended with "Heil Hitler". Ciphers can only be as irregular as the humans who use them, and we are creatures of habit.

Computers were born in World War II: Alan Turing established the principle of the fully programmable computing machine, and John von Neumann invented the electromechanical architecture that made the principle a reality. Grace Hopper then invented the software interfaces, known as computer languages, which allowed us to program the computers quickly and accurately.

Computer languages have gone through at least seven generations. Some would say more, but these seven represent my understanding of computer history.

- **1st generation: machine code.** Instructions were rendered as strings of numbers, supported by a syntax. So 15010414531602430417430400 could mean:

15	01	04	1453	16	02	4304	17	4304	00
Move to register	Number	Four digits	1453	Add to register	Contents of location	4304	Store in location	4304	End

- **2nd generation: assembler.** Instructions were more language-like:
  MOV 1453; ADD #4304; STA #4304
- **3rd generation: programming languages.** FORTRAN, COBOL, ALGOL, BASIC, etc:
  TOTAL = TOTAL + 1453
- **4th generation: multimedia languages.** These were essentially 3rd generation languages with added tools to handle visual and audio elements (e.g. VISUAL BASIC).

- **5th generation: programmable programs.** Spreadsheets, word processors, presentation software, etc.
- **6th generation: apps.** Tasks are done by a series of interrelated specialist apps. The user doesn't need to know the details of how a task is carried out. There is a high level of integration between different task-based software applications (e.g. Microsoft Office).
- **7th generation: "personality apps"** (e.g. Siri, Cortana, Alexa). They are usually referred to as "intelligent personal assistants", but they still have some way to go before can be considered as intelligent, personalised, and more than marginally useful; but if you want to remotely turn off your home heating, Alexa can do it.

Nowadays we are supposed to be close to the **cognitive singularity point**, when the computing capacity of artificial brains will be greater than human brains. At that point, can we expect that either computer language or human language will begin to disappear?

Computer languages taught us a lot about how languages work, mainly from our attempts to describe the texts (programs and apps) we produced. We attempted to describe them as logical structures (how the program was put together), but that told us nothing about how inputs became outputs. So we attempted to describe them as information processes (how the data flowed through the program) but that told us nothing about the structure that controlled the data flow. Eventually we settled on describing them as systems (structures with processes). This, however, is not an easy form of description, but it is inclusive of the what, when, where and how of data management; and, if done right, systemic description is much more informative than structural or processive description. When you are describing your language, remember to treat it as a communication system and not just a coding structure or information process.

ARTLANGS FOR FUN!

Languages created for fun often start out as simple codes – and stay that way. However, some creators take the next step towards a consistent communication system – phonetic, lexical and grammatical functionality. This creates a sufficient conlang.

The next step is cultural – building reasons for the language. At this stage, the language begins to take on its own reality, and non-linguistic features begin to creep in. Flags and maps are designed, histories are written, cultures are defined, in-groups and out-groups established … In the case of nonhuman languages, this can get quite complex, and the conlang moves from sufficient to substantial.

The final step, which few artificial languages achieve, is to build a community of users. This, fortunately, is not part of the module assignment; which is just as well, because it takes years and only occasionally happens.

[1] Johan Huizinga (1950). *Homo ludens: a study of the play element in culture*. Beacon Press: Boston, MA, USA.

[2] In a substantial language, unbridgeable translation gaps seldom or never arise; a sufficient language works well in certain topical areas, less well in others; a sketched-out language gives a set rules and some lexis, and it can produce some utterances. You should aim for a language sufficient to complete your chosen translation.