Indie Postmortem: Reflexive's *Wik & The Fable Of Souls*

By simon hallam

Wik & The Fable Of Souls is a 13MB downloadable PC game, which was released in September 2004. It was developed by Reflexive Entertainment Inc., a company based in Lake Forest, California. Reflexive has been around since late 1997, developing both mainstream PC titles for publishers and smaller self-funded downloadable games.

During 2003 the successful Reflexive Arcade website was created, and by the end of that year the development and publishing of downloadable games became its primary business. With a staff of 11 full-time employees; Reflexive now focuses on the staggered development of two self-funded projects at a time, with a team dedicated to further the development of Reflexive Arcade.

The roots of *Wik* began with what we at Reflexive call a "one-day-quick" prototype, where we regularly invest time in exploring new game ideas and mechanics by building a working, playable prototype. The prototype in question was called "BugEater", written by James C. Smith, and inspired by the arcade classic *Missile Command*. In "BugEater" the player controls several gecko-like creatures stationed on a rock border down the left and right edges of the screen. The object is to protect your grubs from thieving flying bugs by clicking on them causing your gecko to jump in a straight line across the screen and eat the offending bug thus freeing your grub:

The original "BugEater" one-day quick prototype.

What made "BugEater" a lot of fun was a multi-player mode that we termed a MouseParty™. We had eight people crowded around a single PC, with mice plugged into every USB port on every USB hub that we could find laying around the office, resulting in a crazy click-fest where everybody fought to rescue the most grubs before the level finished; it was complete mayhem and a lot of fun! Out of the six or so quick prototypes we short-listed at the time, "BugEater" was chosen for full development.

What Went Right

The Story. One area of the game that remained almost constant from initial conception to the final product was the back story. Originally conceived by myself, the back story revolves around a bunch of cute little grubs that appeared scattered throughout the land one night, each possessing the hypnotic ability to compel other creatures into taking it home and nurturing it. Once back at the creature's home, however, the parasitic grubs would jab its antenna into its heretofore unsuspecting host and steal its life before entering a chrysalis stage. The grub would then emerge from its chrysalis a monstrous creature and instinctively rampage its way back to its own home.

Originally, these grubs fed upon themselves with the new generation literally taking the place of the old, but the older generation found that by sneaking their offspring to some far off place immediately after their once-every-decade mass communal birthing saved their skin by putting

the gruesome burden upon others.

The hero of the story is Wik, a forest dweller who lost his family to the marauding transformed grubs as they worked their way back home after the last birthing. Wik is not about to let the grubs repeat their destruction and so he comes up with a cunning plan. He decides to collect as many grubs as he can find, keeping them all in the magical backpack that he has strapped to his faithful mule Slotham, and journey to return them to their natural home where he plans to release them and restore their cyclical cannibalism.

The story was meant to be told orally in a fairytale style from early in the product's development, but since we could not afford the download space required to ship with an audio narrative, we had to rely on the art and music being something extra special to pull the player along with the text of the story.

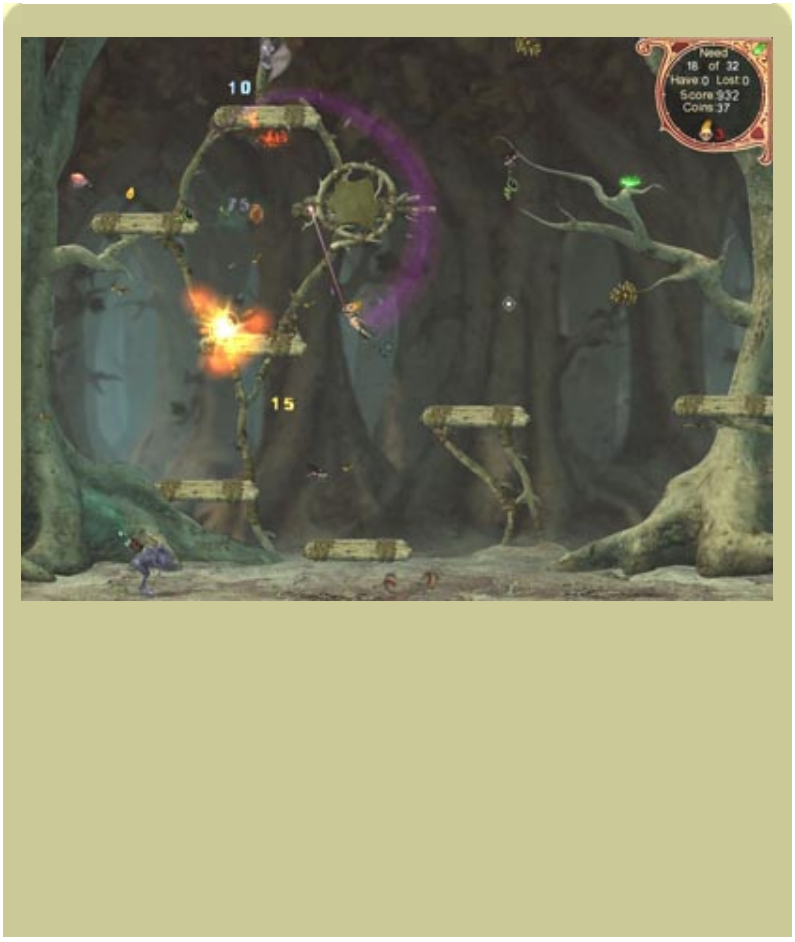
About halfway through development of the project, writer Eric Dellaire was contracted to take the existing story and re-write it as a number of short poems/verses, the final cut of which was not approved until just a few days before the product shipped but we loved the dark and humorous style that Eric came up with, so that all in all, the story element turned out to be a great success.

Composition of Background Art and Foreground Play Elements. Because of several factors including a six month compressed development cycle and 15MB download size limit, we decided the game would take place on a single non-scrolling background frame. Once this decision was made, we began experimenting with art styles for the three environment types that the story called for: a forest, some underground caves, and dark creepy vines.

We figured 70 unique game play levels would be enough to tell the story, and we had plans for another game mode called "challenge mode" which would require about 50 more levels. There was not enough time in our schedule or space in our maximum 15MB download for 120 uniquely painted levels, so we set about developing techniques and tools that would allow us to compose our game play frame from a palette of pre-rendered art pieces.

During pre-production, the lead artist on the project, Jeff McAteer, would take hand-painted elements and models rendered using 3D Studio Max into Photoshop and compose them with filters and effects, building atmospheric concept images. The programming team took note of this and began developing an in-game level editing system that allowed entities called props to be easily placed, rotated, scaled and colored on multiple layers, with basic filters and effects that could be applied as required.

When the game was in full production, the lead artist delegated creation of the final rendered props for level construction to Chung Ho Kan and Zach Young (who also wrote all the music for the game). Level designers Ion Hardie (who also created all the sound effects in the game) and Ernie Ramirez used these art pieces and the new level editing system to create every level in the game. Within the final two weeks of the project, the lead artist took two days going over each level with its respective designer, making minor artistic changes to improve its look. From the initial conception of the environment types through to their final tuning and approval, the process was a smooth one, thanks to good planning up-front and a tool set that performed admirably.



A screenshot of the final *Wik* gameplay .

User Interface Development Tools. Building the user interface in Reflexive games in the past involved our interface designer (Zach Young) creating a mock up in Photoshop to illustrate the overall design and placement of individual interface components. The programmer would then take this mockup and implement it, a process which might take days, while the designer moved on to other tasks. Once implemented, the designer would suggest modifications or sometimes completely re-design the interface once they were finished with whatever task they had been working on in the meantime, and again hand off to the programmer for implementation. While this process worked, it was hardly efficient and often disjointed.

Since our level builder/editing system was turning out to be a great success, it seemed like a natural evolutionary step in its development to create more specialized 'props', and a powerful visual scripting system that would allow us to use this same system to build the games entire user interface. By giving the interface designer the power to place components and add script commands to the various events that would be triggered by them, we essentially took the programmer out of the loop, freeing the two bottlenecks that existed in our previous method. There were still times when the designer requested a new component or feature and had to wait for it to be implemented, but the designer was still free to develop other aspects of the interface.

Building the user interface in this way allowed the artists and designers to get closer to the interface implementation process than they had been able to before. It also allowed them to use all the props, effects and features that exist within the main game on the interface in unexpected ways without tying up limited programming resources nearly as much as in past products.

The Main Player Character. Design of the main character was initially driven by two things -the way he moved in the "BugEater" prototype, and the first draft of the back story. In our story the character had been through a pretty rough time. We wanted to communicate that fact and hoped that people would empathize a little as they read the story, so it was clear that we should try to avoid the reptilian look that the prototypes player character had, since most people have a hard time identifying feelings with reptiles. We figured that trying to meld the look of a small human frame with that of a frog would be an interesting direction, and our lead artist sketched literally hundreds of possible design ideas, some with subtle changes, and others more radical. The end product was a character we called 'Wik', who we felt was a step in the right direction.

The first draft of the back story was basically a tale of revenge, featuring the player character having gone slightly crazy while dealing with the loss of his family, which twisted his character into something of an antihero. As such, many of the early sketches of Wik have a mischievous, some might even call wicked, edge to them. It was not until several months into the games development that Wik's story changed from one of revenge into him searching for the trapped souls of his family, and the softening of his earlier dark intent is reflected in the later visual design of his character.

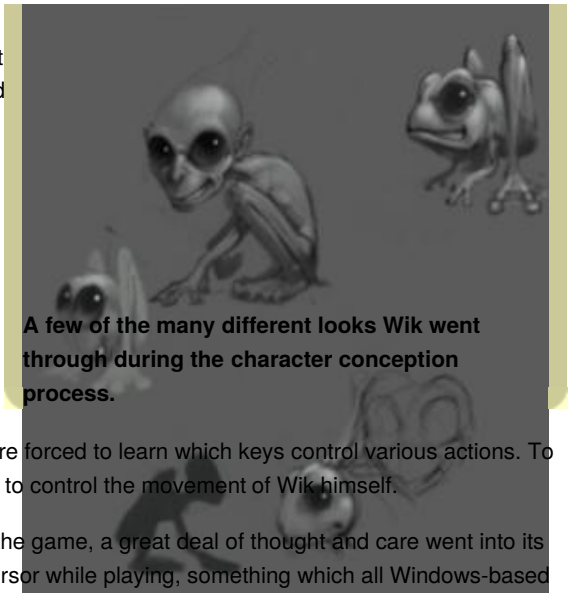
Well into the third month of development, there were serious concerns about the basic play mechanic; it was simply not fun enough as the foundation for the entire game. As in the original prototype, clicking on a bug somewhere on the screen made Wik jump across the entire screen in a straight line, intersecting the point where the player had clicked and snatching the bug along the way. We experimented with obstacles that would force the player to make several jumps around the screen to get to an intended target; we tried automatically grabbing bugs anywhere along Wik's jump path, there were experiments with Wik's long tongue, using it to snatch a bug from the air if it was within a pre-determined maximum range instead of him jumping, and a few other things that didn't seem to hit the mark either.

It was Brian Fisher, programmer and physics guru, who came up with the idea of applying gravity to Wik's movement. The producer was initially unconvinced that gravity was the answer -it was interesting placing platforms around a level and jumping around, but it seemed like too radical a change to make given the limited development time remaining, and by itself it didn't add the extra spark of something we were looking for. After a couple of days experimenting and hopping around though, we struck upon the idea of seeing what would happen if Wik's long tongue stuck to platforms in a way that would allow him to dangle and swing, latching onto things mid jump. Even though the first version of this was implemented quite roughly, it was immediately apparent that we had hit upon something special. The addition of gravity marked a monumental turning point in Wik's development.

Changing the existing game to take advantage of this new jump and swing play mechanic would clearly take extra time to develop properly and balance; it would also mean that every level the designers had built to date would need to be scrapped. Deciding to commit to this new direction was not something to be taken lightly, but the game was clearly much more fun than everything else we had tried, and the jump-swinging introduced an element of skill that players could learn and improve over time.

It took almost six weeks to get Wik and a small set of reference game levels balanced with the jumping and swinging play mechanic, but by the end of this period

we had added other features, such as the ability to control Wik dangling from his tongue as though he is on a playground swing, simply by moving the mouse from left to right. His latched tongue also now collected bugs that flew into it like fly paper, and once those bugs were pulled back in his mouth Wik could spit them out shotgun style at great velocity, knocking other bugs out of the air. To tune and control all of Wik's new abilities, we created something called the "Wik style sheet", which is a list of 86 easily modifiable in-game editable property fields. This addition meant that we no longer had to re-compile the project source code each time we made a change to Wik's behavior.



A few of the many different looks Wik went through during the character conception process.

Mouse-Only Controls. A primary goal for any Reflexive game released into the current downloadable game space is that it should be possible to control every aspect of the game using only the mouse. We learned that a large percentage of downloadable game players are immediately turned off from any game where they are forced to learn which keys control various actions. To that end, the mouse is used to control every aspect of Wik's user interface, and also, to control the movement of Wik himself.

While Wik's player control model may seem obvious to most of the people who play the game, a great deal of thought and care went into its design. The player directly controls the position of a fairly standard looking mouse cursor while playing, something which all Windows-based computer users are very familiar with, as in Windows, the left mouse button performs the primary action, which in this case means Wik either launching his tongue out as closely to the current mouse cursor position as possible, or him spitting whatever is currently in his mouth. The right Mouse button performs the secondary action, which follows the model used in many console games where the secondary action is to jump, here Wik will attempt to jump as close to the mouse cursor as he can reach.

But what does jumping, or spitting, or latching Wik's tongue as closely to the "current mouse cursor position" really mean when he can only jump horizontally about as far as he is tall, or launch his tongue less than 1/3 the full height of the screen, or spit multiple objects of differing masses? Well, it means that there is a lot more internal calculation and prediction happening than people generally realize, all of which is designed to make Wik feel like he is doing 'the right thing' from the players perspective. For example, when the player clicks the jump button, Wik will normally calculate a jump trajectory that has the center of his mass traveling through the center of the mouse cursor, but when he is stood on a platform and the player clicks for a jump anywhere close to the edge of that platform, Wik does just about everything in his power to jump without falling off the edge of that platform.

Another example of the interpretation of what the player "probably wants to do" includes tongue latch click positions, where the literal player input may be heavily modified so that as Wik jumps through the air, he searches for an optimum swing position around the click location rather than risk missing a latch altogether and possibly falling to a place where he loses a level because he fell into a pit of scorpions. The game is designed to do as much as it can to help the player have a positive experience, erring on the side of helping perform an action successfully rather than blindly accepting that the player must have meant to fall to his doom when he clicked just 4 pixels outside of a tongue latch area.

What went Wrong

Initial play mechanic. During its third month of development, we had still not found what made the play mechanic we had in place intrinsically fun. The designers had serious concerns regarding how to build levels which were anything but visually different from one another, and none of the experiments designed to add depth and fun to the play mechanic seemed to be working out. By the end of that third month we finally decided to get radical and re-design the play mechanic from the ground up.

Whether or not the original "BugEater" prototype, which *Wik* was founded upon, is truly fun as a single player game has become a subject of some controversy within Reflexive. The prototype's original creator believes firmly that it demonstrates a solid foundation for a single player experience, whereas the producer on *Wik* feels that we were somewhat blinded by our experiments playing "BugEater" in MouseParty™ mode, leading to the expectation that embellishing the original play mechanic with a few extra features would lead to a fun single player mode.

Thanks to the ingenuity and creativity of the team, *Wik* turned into a product that we are all very proud of, but for more than half of its originally planned development time it was a completely different and much less fun game. Couldn't we have handled that better?

Probably the biggest lesson to be learned from this product is, "if you're responsible for ensuring it's fun, you need to own the concept 100%". This means that when transitioning into full development, either the person who originally conceived the prototype should become its ultimate lead programmer/producer, or whoever inherits those roles should take them over completely, believing passionately in either the original play mechanic, or accepting that it must be changed into something he/she can believe in.

In the case of *Wik*, the producer inherited the "BugEater" prototype from somebody who was ultimately not part of the *Wik* development team, and while believing that playing "BugEater" in MouseParty™ mode was a lot of fun, he did not believe passionately in the single player play mechanic. But rather than accept immediately that the single player mechanic would have to be changed radically to become something he considered "fun", development progressed in a way that took it on faith that in time he would begin to see and understand the vision of the prototype's original creator, something which never happened.

The grub parasitic cycle.

One of the ideas designed to embellish the early play mechanic was termed the "dynamic stinky cheese". At the time, Wik did not pick up grubs and toss them into Slotham's magic backpack the way he does in the finished product. Instead, he attached a large chunk of "stinky cheese" onto the back of his faithful mule Slotham. This stinky cheese was the only thing that Wik had discovered which would lure the Grubs into following Slotham rather than executing their own diabolical plan.

The smell of the stinky cheese was represented by a green colored particle system, the movement of which was managed by a surprisingly complex real time fluid dynamics simulation. As Wik moved around the level, the cheesy smell would be carried along by the movement of air currents around him, creating swirls and vortices in his wake.

How could the green grubs resist this pungent yet graceful green cloud? Well, they couldn't - that was sort of the whole point of the exercise. Once Wik moved in such a way as to waft several of the green particles over the grubs' antenna, they were forever under the spell of its stench (these grubs really like cheese). A thin tendril of particles would then curve from the cheese on Slotham's back to each grub that had ever 'smelled' it. The grubs would then do everything in their power to move closer to the cheese and follow Slotham off the level, counting towards the players score as a grub captured.

Intelligent tutorial system. A crucial element to the successful completion of levels in this product is learning how to make Wik swing effectively. "To swing is king, if you do it right Wik will sing." During development, effective tongue latch swinging was something that most of the people working on the project picked up naturally, but there were a couple of people in the office who had difficulty getting a good swing action going, and then timing its release so that Wik would fly off in the direction they had intended. During focus group testing, we also noticed several people were having difficulty.

We set about solving this swinging difficulty, and other issues people were having by developing an in-game tutorial system. During the first several levels in the game, we made message boxes and arrows appear which would point out locations to latch Wik's tongue to, or grab something, and describe how to manipulate the mouse to complete a very specific goal. This was done in a linear fashion, with every event pre-scripted. During subsequent focus group testing, we found that people who initially had difficulty could complete these tutorial sections, but when left to their own devices in the remainder of the game they often fell back on old habits, apparently forgetting what they had learned in the tutorial. We also noticed that those people who had previously quickly become intermediate players were now frustrated and bored by the tutorial levels.

So we decided to create a new set of tutorial levels, where there were several basic goals to be completed on each and the scripts we had set up would be more intelligent about what help they offered, and would only offer it if the player appeared to be unsure what their next goal was. Our hope was that intermediate to advanced players would be able to complete a tutorial level quickly and without being annoyed by message boxes demanding that they perform each of a set of individual tasks right now, while beginners would better hone their skills through repetition before the system decided to tell them about the next goal.

Focus group testing indicated that we had improved previous difficulties greatly. Intermediate to advanced players were able to find interesting new ways to complete the tutorial levels in ways that engaged them, and most of the beginners who were having problems before now left the tutorial levels with better skills.

There were still problems even with the new tutorial system, as we found that there were some players who were able to complete the levels by some measure of fluke and were struggling on much later levels in the game because they had managed to get by without learning a particular skill thoroughly enough. We decided that we had probably gone far enough with the tutorial system, and developed a "you can't please all of the people all of the time" attitude.

Since *Wik*'s release, however, we struck upon the idea of having a third tutorial system that would work as a quiet watchdog most of the time, offering guidance on how to improve a specific technique only when it notices an area of weakness. Even if this system could only monitor and offer advise on the players tongue latch swing technique, we think it would help those few people who play *Wik* but don't really get into the swing of it (pun intended) have a more enjoyable experience.

Omission of bosses. Our early game plan called for four boss characters which *Wik* would encounter and battle through the game. The Hornet Queen would appear towards the end of the forest level set, the Spider Boss would flash her fangs towards the end of the caves, a giant Scorpion would appear towards the end of the creepy vines level set, and the end of game ultimate uber-boss was the Lord Of The Grubs, a fearsome creature guarding the gates of the land where the wicked little grubs had originated.

The Hornet Queen and the Spider Boss were both modeled, had basic texturing and were fully animated before the difficult decision was made to cut them and the other two boss creatures from the game. We simply did not have enough time in the development schedule to implement and balance them after taking time to re-design the core play mechanic.

The addition of gravity also meant that some of the ways we were planning for the bosses to chase Wik around the level did not make sense any more. The spider boss was designed to follow Wik as he jumped and landed at positions stationed around the inside edge of the game play frame. She would land directly on top of Wik, matching his orientation and trying to bury those wicked fangs deep in our hero before there was chance for Wik to jump away. While the initial experiments looked pretty cool, once Wik was jumping onto platforms and swinging around in levels, the way the Spider Boss moved seemed odd and no longer fit well with the rest of the game.

OpenGL compatibility problems. Post release we had a small percentage of people reporting strange rendering issues and lockups (which seemed to be related) while playing the game. Extrapolating based on the number of people reporting issues; we estimated that between 8% and 12% of people who downloaded the trial were experiencing compatibility problems.

We take compatibility testing seriously on all our products, and while we had not sent this product to an external testing facility to compile a detailed report, we had tested it on the 25 machines we have in-house, which includes our development machines and two rooms full of machines of varying ages with every version of the Windows operating system we support. There were also around 20 external beta testers who played the game on their own machines, so by the time we released the game we thought we had most of the major issues resolved.

Once we realized that a significant number of users were experiencing these serious problems, we set about finding machines we could get in-house and begin debugging. Within a couple of days we had a few machines from friends and relatives that were exhibiting the problems described in tech support emails and on the www.WikGame.com forum. We quickly learned that the issues were related to our use of OpenGL, and an operating system manufacturer's automatic update process assuring the end user that they had the very latest version of drivers available for their video card, when in fact they did not.

An early painting demonstrates the look that we were aiming for and gives a sense of the game play; the end product was remarkably similar.

Before shipping *Wik*, we had planned to rely upon trusty SDL to handle initialization and shutdown of OpenGL, but since the game worked fine on every machine we had tested pre-release it didn't really seem like a worthwhile investment of resources. There was some ambiguity as to whether using SDL would really have helped the specific problems we were seeing, and since Reflexive is not in the business of providing operating system or video card driver support, we decided to fall back on good old trusty software rasterizing.

Wik's rendering pipeline relies upon being able to display a great many pre-rendered 2D bitmap images each game frame. These bitmaps are

arbitrarily colored, scaled, rotated and blended together. We had been considering building a rendering pipeline based upon AGG for some time - programmer and CEO Lars Brubaker is a huge proponent of open source software and libraries, so he set about modifying and integrating AGG to be the core of *Wik*'s new software rasterizer. After about two weeks of work, which included the necessary calming down of some of the wilder particle system effects that the previous hardware dependent implementation had afforded us, and a great deal of modifications to AGG, *Wik* was re-launched with its 3D-capable video card and OpenGL driver minimum specification removed.

Conclusion

Wik & The Fable Of Souls has received wide critical acclaim, with rave reviews, warm letters from happy purchasers, and now having made the finals of the Independent Games Festival competition, it is difficult to imagine being happier about the way people have received *Wik*.

And yet to date, the product has underperformed commercially. Perhaps we don't entice enough downloads via marketing, or perhaps only the people who really get into the product are the ones sharing their rave opinions, maybe it just takes a little longer for a product of this type to be fully accepted by the downloadable games community? Whatever the reason, Reflexive believes that this is the future of downloadable gaming, an out-of-the-box creative experience that lays somewhere between the hordes of downloadable puzzle games and the multi-million dollar epics funded by publishers.

Try-before-you-buy downloadable games is not an easy market for a developer to survive in, making your game look better than it really is on the box art or in the print ad will not help here, there is no marketing department to hide behind, your prospective purchaser gets up to 90 minutes of one-on-one time with your work before they decide if it's worthy of their hard-earned money or not, and if they decide not, there are any of a hundred more downloadable products they can be playing within a couple of minutes via broadband. To that end, Reflexive will continue to produce the commercial hits like the *Ricochet* series, and *Big Kahuna Reef* that it always has, but every once in a while we will also put something out there like *Wik* that breaks the mold and tries to push the boundaries of downloadable game players' expectations.

You can find out more about *Wik & The Fable Of Souls* at www.WikGame.com.

Game Data

Target platform: Pentium 733 with 128MB, Windows based OS.
Length of development cycle: 9 Months
Budget: $350,000
Internal team members: 9
External contractors: 3
Development workstations: 1.5GHz to 2.7GHz PC's running Windows XP Pro
Development tools: Visual C++ 6, 3D Studio Max 6, ZBrush, Photoshop, Cool Edit Pro, Korg Triton Workstation
External libraries used: Heavily modified Anti Grain Geometry, OGG, Zlib.
Size of project source materials: 539MB (uncompressed)
Size of final product: 13.9MB
Release Date: August 9th 2004

Return to the full version of this article